Initialise!

Insights into the startup process and how to secure it

Marko Jung

Global Head of Information Security Operations

 *m@mju.ng*

 *@mjung*

 *fb.com/markohjung*

HOW TO "READ" FM TUNER SPECIFICATIONS

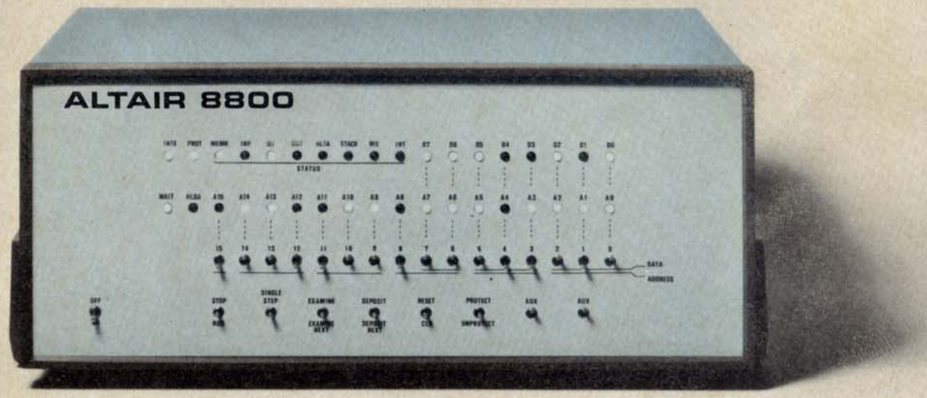
Popular Electronics

WORLD'S LARGEST-SELLING ELECTRONICS MAGAZINE JANUARY 1975/75¢

PROJECT BREAKTHROUGH!

World's First Minicomputer Kit to Rival Commercial Models...

"ALTAIR 8800" SAVE OVER $1000



ZIFF-DAVIS PUBLISHING COMPANY

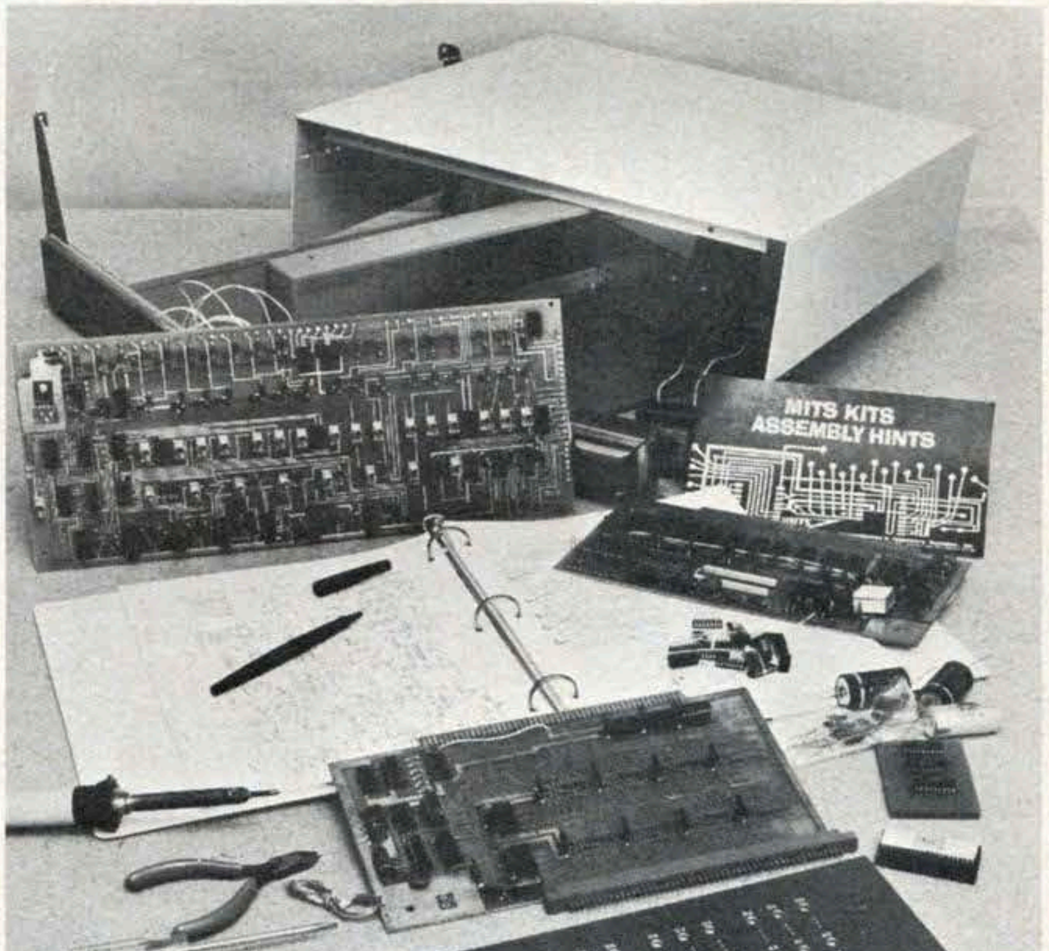
ALSO IN THIS ISSUE:

- An Under-$90 Scientific Calculator Project
- CCD's—TV Camera Tube Successor?
- Thyristor-Controlled Photoflashers

MITS

BUILDING YOUR OWN COMPUTER WON'T BE A PIECE OF CAKE.

(But, we'll make it a rewarding experience.)



Chances are you won't be able to assemble the *Altair 8800 Computer* in an hour or two. But, that's only because the *Altair* is a real, full-blown computer. It's not a demonstration kit.

The *Altair Computer* is fast, powerful, and flexible. Its basic instruction cycle time is 2 microseconds. It can directly address 256 input and 256 output devices **and** up to 65,000 words of memory.

Thanks to buss orientation and wide selection of interface cards the *Altair 8800* requires almost no design changes to connect with most external devices. Up to 15 additional cards can be added inside the main case.

BUILDING YOUR OWN COMPUTER WON'T BE A PIECE OF CAKE.

(But, we'll make it a rewarding experience.)

Chances are you won't be able to assemble the *Altair 8800 Computer* in an hour or two. But, that's only because the *Altair* is a real, full-blown computer. It's not a demonstration kit.

The *Altair Computer* is fast, powerful, and flexible. Its basic instruction cycle time is 2 microseconds. It can directly address 256 input and 256 output devices **and** up to 65,000 words of memory.

Thanks to buss orientation and wide selection of interface cards the *Altair 8800* requires almost no design changes to connect with most external devices. Up to 15 additional cards can be added inside the main case.

The *Altair Computer* kit is about as difficult to assemble as a desktop calculator. If you can handle a soldering iron and follow simple instructions, you can build a computer.

You see, at *MITS,* we want your experience with our kits to be rewarding. That's why we take such pains to write an accurate, straight-forward assembly manual. One that you follow step-by-step. (We leave nothing to the imagination.)

Some electronic kit companies are experts at cutting the corners. They promise you the sky and deliver a box full of surplus parts and a few pages of faded instructions run off on their copying machine.

We're experts at **not** cutting the corners. Our *Altair Computer* has been designed for both the hobby and the industrial market.

PRICES: Altair Computer Kit with complete assembly instructions **$439.00**
Assembled Altair Computer **$621.00**


ALTAIR 8800 COMPUTER


ALTAIR PAPER TAPE LOADER
BILL GATES
17 BYTES
JOSH BENSADON +
~~14~~ BYTES
12

JUST 8 BITS, WHAT CAN YOU DO WITH?

Monday, 20 March 2017

Doing it in less than Bill Gates. Loading Microsoft 4K BASIC on ALTAIR 8800 with Papertape

Two years ago, I demonstrated the loading of Micro-Soft's first software product on their first platform. If you didn't already know, it was BASIC for the ALTAIR 8800 computer. The story is well known in the vintage computer community. It was a real success for Bill Gates and Paul Allen and they are nothing less than artists!

In preparation for the demonstration at World of Commodore 2014, I found there were multiple bootstrap programs. It depended on hardware. The manual has two, a 21 byte and a 20 byte version. Over the course of a month, I studied the boot strap and tape in detail and found it really just loads another bootstrap program with better capabilities. It may seem that 20 bytes is not a lot, but there should be a warning in the manual "fingers will get sore after repeated use of the small switches on the ALTAIR". Further, each byte in turn lead to a greater probability of making an error.

The show went well, my friend Jeff Brown made a great video of the event "A re-tracing of how Paul Allen loaded BASIC on the MITS Altair 8800 from paper tape". I really must thank him for the idea of loading 4K BASIC! He saw my ALTAIR on a visit where we repaired his PET 2001 and suggested it. After he released his youtube video, I stumbled upon another video called "Bill Gates talks about Microsoft and the Altair 8800". At 3:26 into the video, Bill talks about the bootstrap loader and says Paul wrote the first loader in 46 bytes and he (Bill) later wrote it in 17 bytes. I thought to myself, yes, 17 bytes is better than 20 bytes... especially if you are repeatedly demonstrating it. I played with the idea of figuring out those 17 bytes then set out to make it as small as possible. Well, that's when I got it down to just 14 bytes and now down further (with some encouragement) to 12 bytes!

About Me

Josh B

Follow 5

View my complete profile

Numeral Systems

```
11 101 010   Binary
 3   5   2   Octal
         234 Decimal
```

12 Byte 2SIO Bootstrap Loader

```
041   LXI H,     ;Initialise Memory pointer to 0x01DB
333   IN 1       ;Read UART data register (skip the status register)
001
276   CMP M      ;Compare with data stored at memory pointer
312   JZ 0001    ;Loop back if data is the same as stored
001
000
053   DCX H      ;When it's a different byte, decrement the memory pointer
167   MOV M,A    ;and save in memory
303   JMP 0001   ;Loop back for the next byte.
001
000
```

Source: http://just8bits.blogspot.com/2017/03/doing-it-in-less-than-bill-gates.html

20 Byte 2SIO Bootstrap Loader

```
000:041 256 017 061 022 000 333 000
010:017 330 333 001 275 310 055 167
020:300 351 003 000
```


MEMSIZ?

Apple Introduces the First Low Cost Microcomputer System with a Video Terminal and 8K Bytes of RAM on a Single PC Card.

The Apple Computer. A truly complete microcomputer system on a single PC board. Based on the MOS Technology 6502 microprocessor, the Apple also has a built-in video terminal and sockets for 8K bytes of on-board RAM memory. With the addition of a keyboard and video monitor, you'll have an extremely powerful computer system that can be used for anything from developing programs to playing games or running BASIC.

Combining the computer, video terminal and dynamic memory on a single board has resulted in a large reduction in chip count, which means more reliability and lowered cost. Since the Apple comes fully assembled, tested & burned-in and has a complete power supply on-board, initial set-up is essentially "hassle free" and you can be running within minutes. At $666.66 (including 4K bytes RAM!) it opens many new possibilities for users and systems manufacturers.

You Don't Need an Expensive Teletype.

Using the built-in video terminal and keyboard interface, you avoid all the expense, noise and maintenance associated with a teletype. And the Apple video terminal is six times faster than a teletype, which means more throughput and less waiting. The Apple connects directly to a video monitor (or home TV with an inexpensive RF modulator) and displays 960 easy to read characters in 24 rows of 40 characters per line with automatic scrolling. The video display section contains its own 1K bytes of memory, so all the RAM memory is available for user programs. And the Keyboard Interface lets you use almost any ASCII-encoded keyboard.

The Apple Computer makes it possible for many people with limited budgets to step up to a video terminal as an I/O device for their computer.

No More Switches, No More Lights.

Compared to switches and LED's, a video terminal can display vast amounts of information simultaneously. The Apple video terminal can display the contents of 192 memory locations at once on the screen. And the firmware in PROMS enables you to enter, display and debug programs (all in hex) from the keyboard, rendering a front panel unnecessary. The firmware also allows your programs to print characters on the display, and since you'll be looking at letters and numbers instead of just LED's, the door is open to all kinds of alphanumeric software (i.e., Games and BASIC).

8K Bytes RAM in 16 Chips!

The Apple Computer uses the new 16-pin 4K dynamic memory chips. They are faster and take ¼ the space and power of even the low power 2102's (the memory chip that everyone else uses). That means 8K bytes in sixteen chips. It also means no more 28 amp power supplies.

The system is fully expandable to 65K via an edge connector which carries both the address and data busses, power supplies and all timing signals. All dynamic memory refreshing for both on and off-board memory is done automatically. Also, the Apple Computer can be upgraded to use the 16K chips when they become available. That's 32K bytes on-board RAM in 16 IC's—the equivalent of 256 2102's!

A Little Cassette Board That Works!

Unlike many other cassette boards on the marketplace, ours works every time. It plugs directly into the upright connector on the main board and stands only 2" tall. And since it is very fast (1500 bits per second), you can read or write 4K bytes in about 20 seconds. All timing is done in software, which results in crystal-controlled accuracy and uniformity from unit to unit.

Unlike some other cassette interfaces which require an expensive tape recorder, the Apple Cassette Interface works reliably with almost any audio-grade cassette recorder.

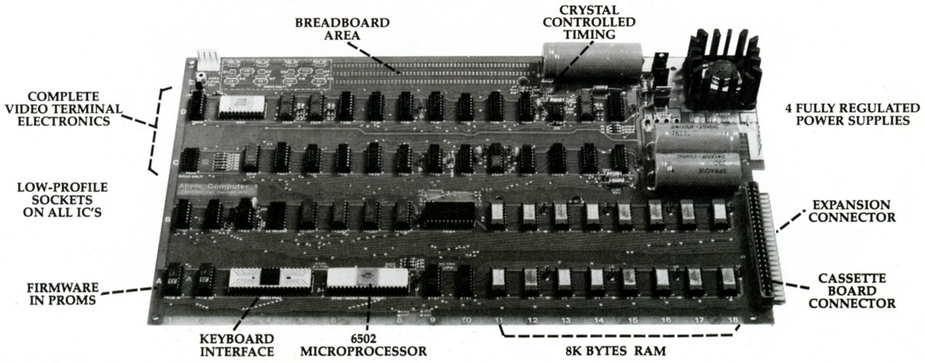
Software:

A tape of **APPLE BASIC** is included free with the Cassette Interface. Apple Basic features immediate error messages and fast execution, and lets you program in a higher level language immediately and without added cost. Also available **now** are a dis-assembler and many games, with many software packages, (including a macro assembler) in the works. And since our philosophy is to provide software for our machines free or at minimal cost, you won't be continually paying for access to this growing software library.

The Apple Computer is in stock at almost all major computer stores. (If your local computer store doesn't carry our products, encourage them or write us direct). **Dealer inquiries invited.**

Byte into an Apple $666.66*

* includes 4K bytes RAM



APPLE Computer Company • 770 Welch Rd., Palo Alto, CA 94304 • (415) 326-4248

CIRCLE NO. 7 ON INQUIRY CARD

Apple Introduces the First Low Cost Microcomputer System with a Video Terminal and 8K Bytes of RAM on a Single PC Card.

The Apple Computer. A truly complete microcomputer system on a single PC board. Based on the MOS Technology 6502 microprocessor, the Apple also has a built-in video terminal and sockets for 8K bytes of on-board RAM memory. With the addition of a keyboard and video monitor, you'll have an extremely powerful computer system that can be used for anything from developing programs to playing games or running BASIC.

Combining the computer, video terminal and dynamic memory on a single board has resulted in a large reduction in chip count, which means more reliability and lowered cost. Since the Apple comes fully assembled, tested & burned-in and has a complete power supply on-board, initial set-up is essentially "hassle free" and you can be running within minutes. At $666.66 (including 4K bytes RAM!) it opens many new possibilities for users and systems manufacturers.

You Don't Need an Expensive Teletype.

Keyboard Interface lets you use almost any ASCII-encoded keyboard.

The Apple Computer makes it possible for many people with limited budgets to step up to a video terminal as an I/O device for their computer.

No More Switches, No More Lights.

Compared to switches and LED's, a video terminal can display vast amounts of information simultaneously. The Apple video terminal can display the contents of 192 memory locations at once on the screen. And the firmware in PROMS enables you to enter, display and debug programs (all in hex) from the keyboard, rendering a front panel unnecessary. The firmware also allows your programs to print characters on the display, and since you'll be looking at letters and numbers instead of just LED's, the door is open to all kinds of alphanumeric software (i.e., Games and BASIC).

8K Bytes RAM in 16 Chips!

The Apple Computer uses the new

ble. That's 32K bytes on-board RAM in 16 IC's—the equivalent of 256 2102's!

A Little Cassette Board That Works!

Unlike many other cassette boards on the marketplace, ours works every time. It plugs directly into the upright connector on the main board and stands only 2" tall. And since it is very fast (1500 bits per second), you can read or write 4K bytes in about 20 seconds. All timing is done in software, which results in crystal-controlled accuracy and uniformity from unit to unit.

Unlike some other cassette interfaces which require an expensive tape recorder, the Apple Cassette Interface works reliably with almost any audio-grade cassette recorder.

Software:

A tape of **APPLE BASIC** is included free with the Cassette Interface. Apple Basic features immediate error messages and fast execution, and lets you program in a higher level lan-

Photo: Ed Uthman - originally posted to Flickr as Apple I Computer

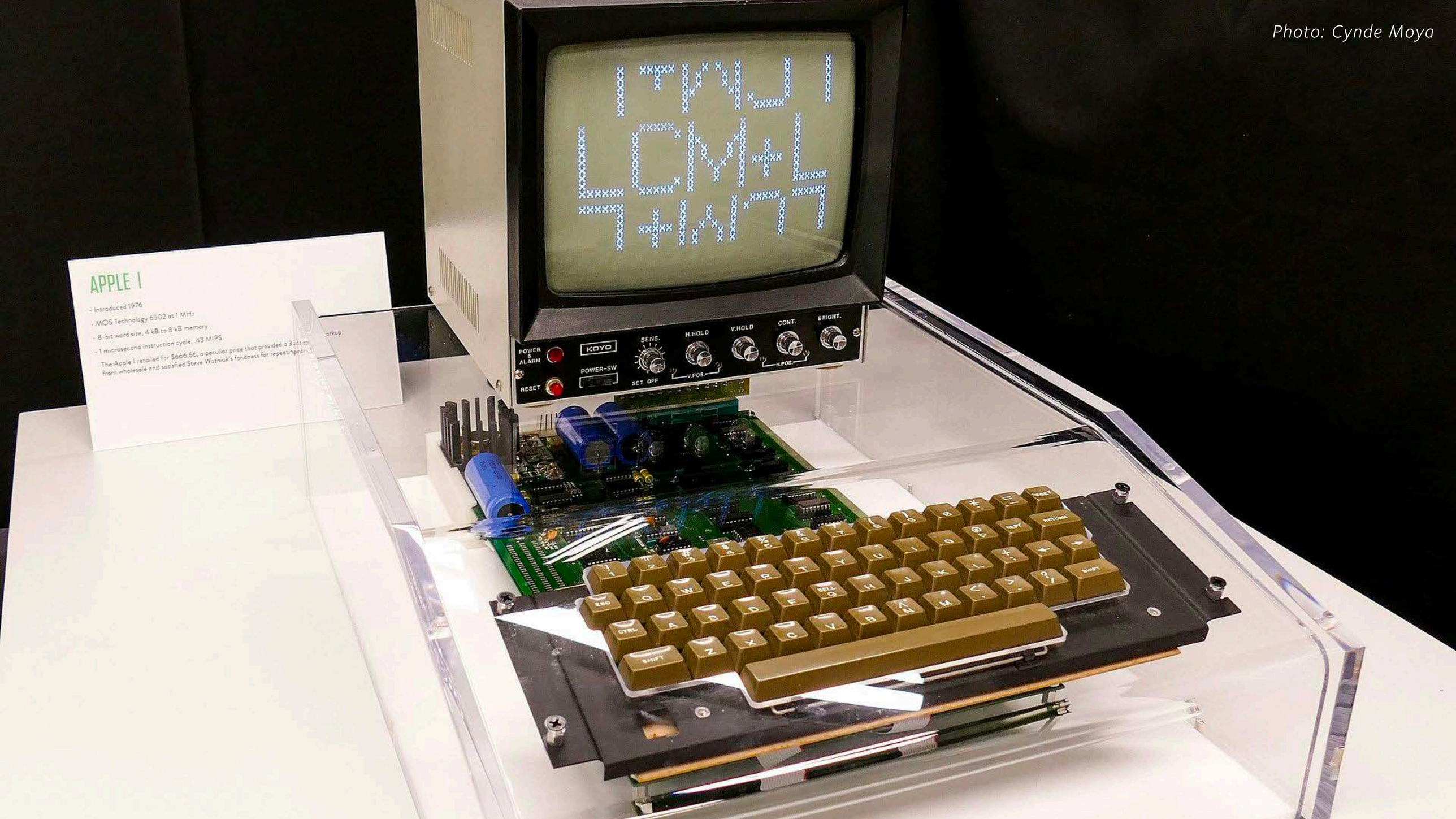
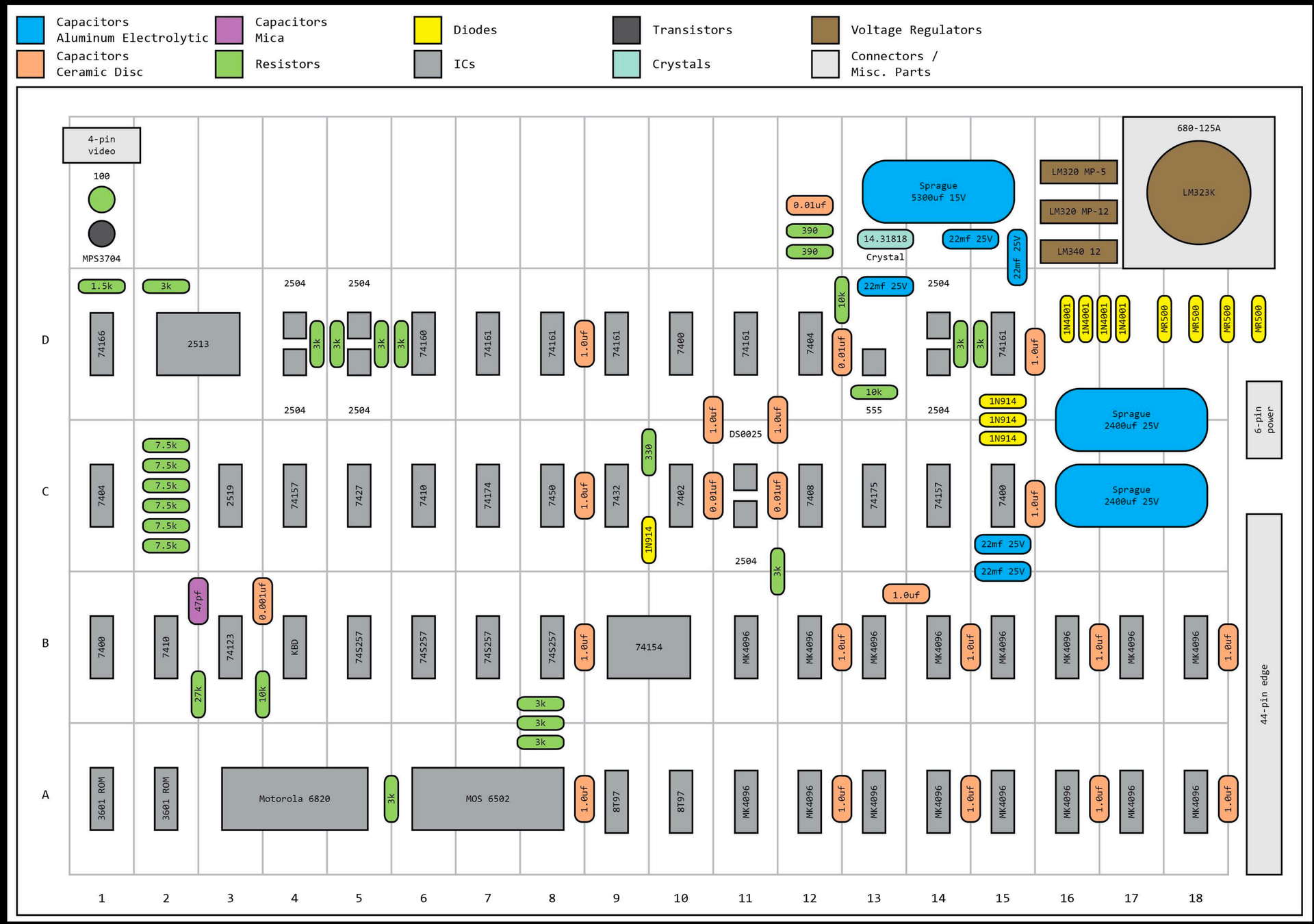
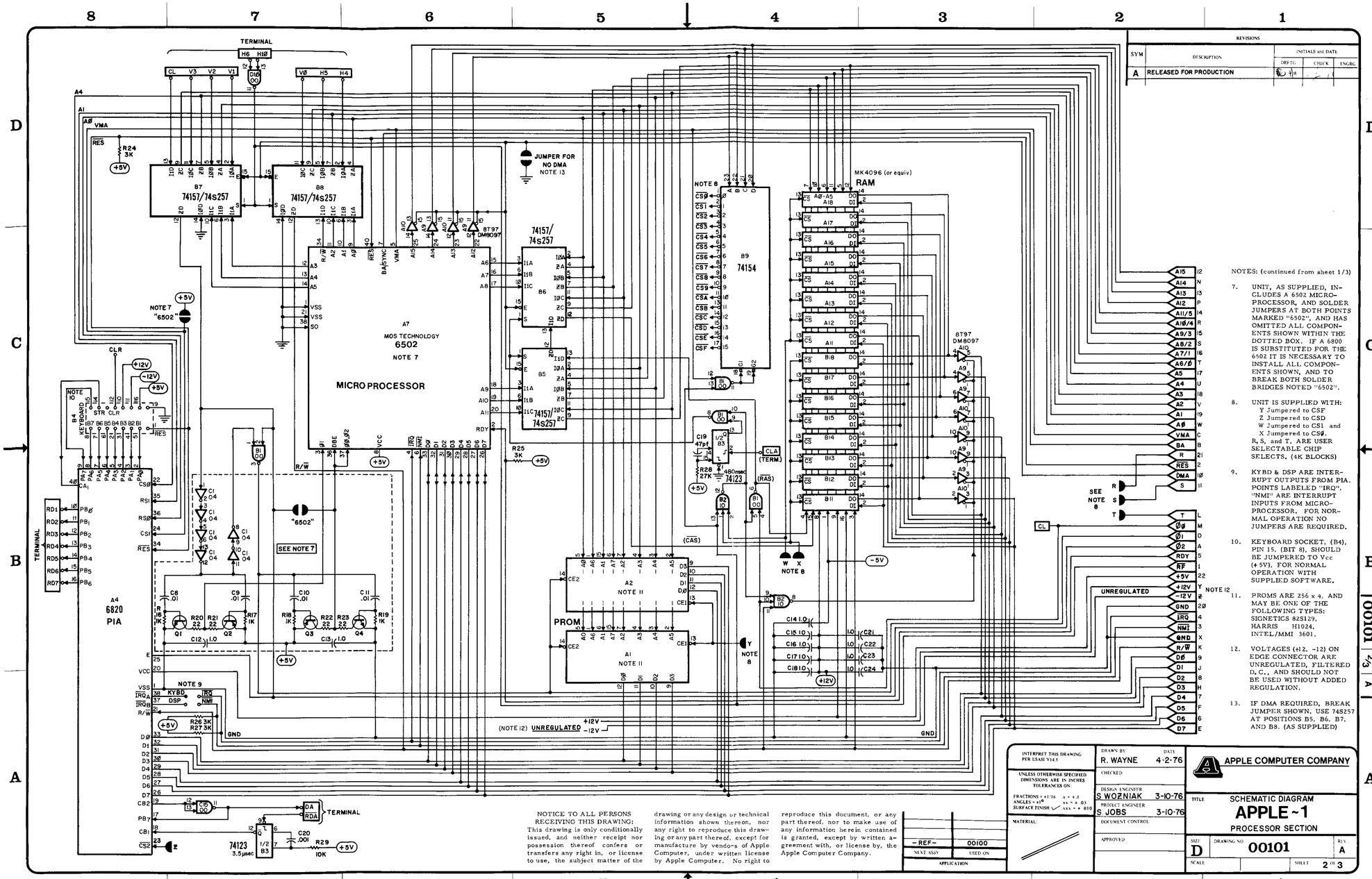


Photo: Cynde Moya

Photo: Binarysequence



Source https://github.com/macmade/XS-Computer-One/



NOTES: (continued from sheet 1/3)

7. UNIT, AS SUPPLIED, INCLUDES A 6502 MICROPROCESSOR, AND SOLDER JUMPERS AT BOTH POINTS MARKED "6502", AND HAS OMITTED ALL COMPONENTS SHOWN WITHIN THE DOTTED BOX. IF A 6800 IS SUBSTITUTED FOR THE 6502 IT IS NECESSARY TO INSTALL ALL COMPONENTS SHOWN, AND TO BREAK BOTH SOLDER BRIDGES NOTED "6502".

8. UNIT IS SUPPLIED WITH:
   Y Jumpered to CSF
   Z Jumpered to CSD
   W Jumpered to CS1 and
   X Jumpered to CS#
   R, S, and T, ARE USER SELECTABLE CHIP SELECTS. (4K BLOCKS)

9. KYBD & DSP ARE INTERRUPT OUTPUTS FROM PIA. POINTS LABELED "IRQ", "NMI" ARE INTERRUPT INPUTS FROM MICROPROCESSOR. FOR NORMAL OPERATION NO JUMPERS ARE REQUIRED.

10. KEYBOARD SOCKET, (B4), PIN 15, (BIT 8), SHOULD BE JUMPERED TO Vcc (+5V), FOR NORMAL OPERATION WITH SUPPLIED SOFTWARE.

11. PROMS ARE 256 x 4, AND MAY BE ONE OF THE FOLLOWING TYPES: SIGNETICS 82S129, HARRIS H1024, INTEL/MMI 3601.

12. VOLTAGES (+12, -12) ON EDGE CONNECTOR ARE UNREGULATED, FILTERED D.C., AND SHOULD NOT BE USED WITHOUT ADDED REGULATION.

13. IF DMA REQUIRED, BREAK JUMPER SHOWN. USE 74S257 AT POSITIONS B5, B6, B7, AND B8. (AS SUPPLIED)

REVISIONS				
SYM	DESCRIPTION	INITIALS and DATE DRFTG	CHECK	ENGRG
A	RELEASED FOR PRODUCTION	4/8		

INTERPRET THIS DRAWING PER USASI Y14.5

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES ON:
FRACTIONS ± 1/16 .x ± .1
ANGLES ± 1° .xx ± .03
SURFACE FINISH √ .xxx ± .010

MATERIAL

- REF -	00100
NEXT ASSY	USED ON
APPLICATION	

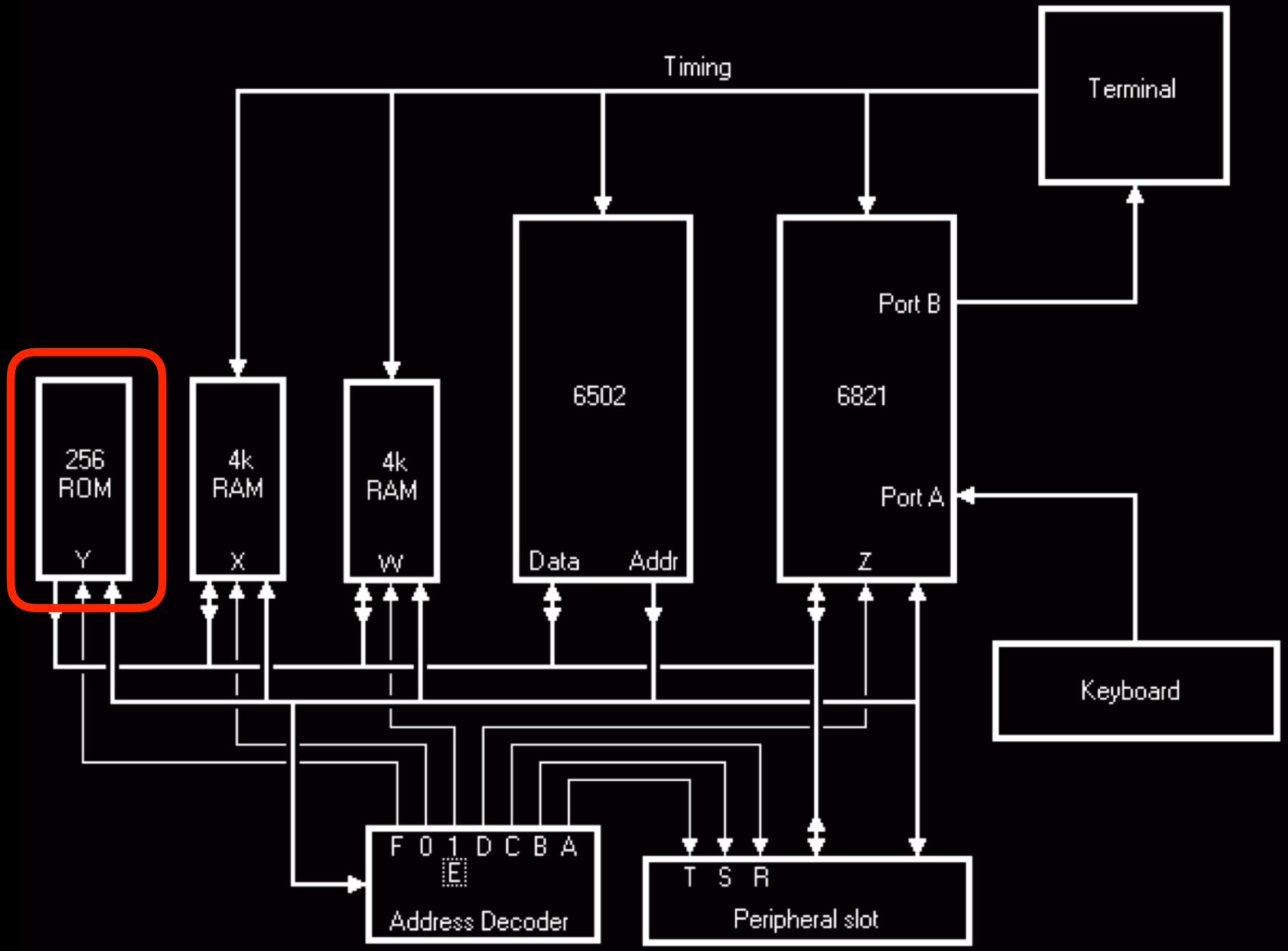
DRAWN BY	R. WAYNE	DATE 4-2-76
CHECKED		
DESIGN ENGINEER	S WOZNIAK	3-10-76
PROJECT ENGINEER	S JOBS	3-10-76
DOCUMENT CONTROL		
APPROVED		

APPLE COMPUTER COMPANY

TITLE SCHEMATIC DIAGRAM
APPLE-1
PROCESSOR SECTION

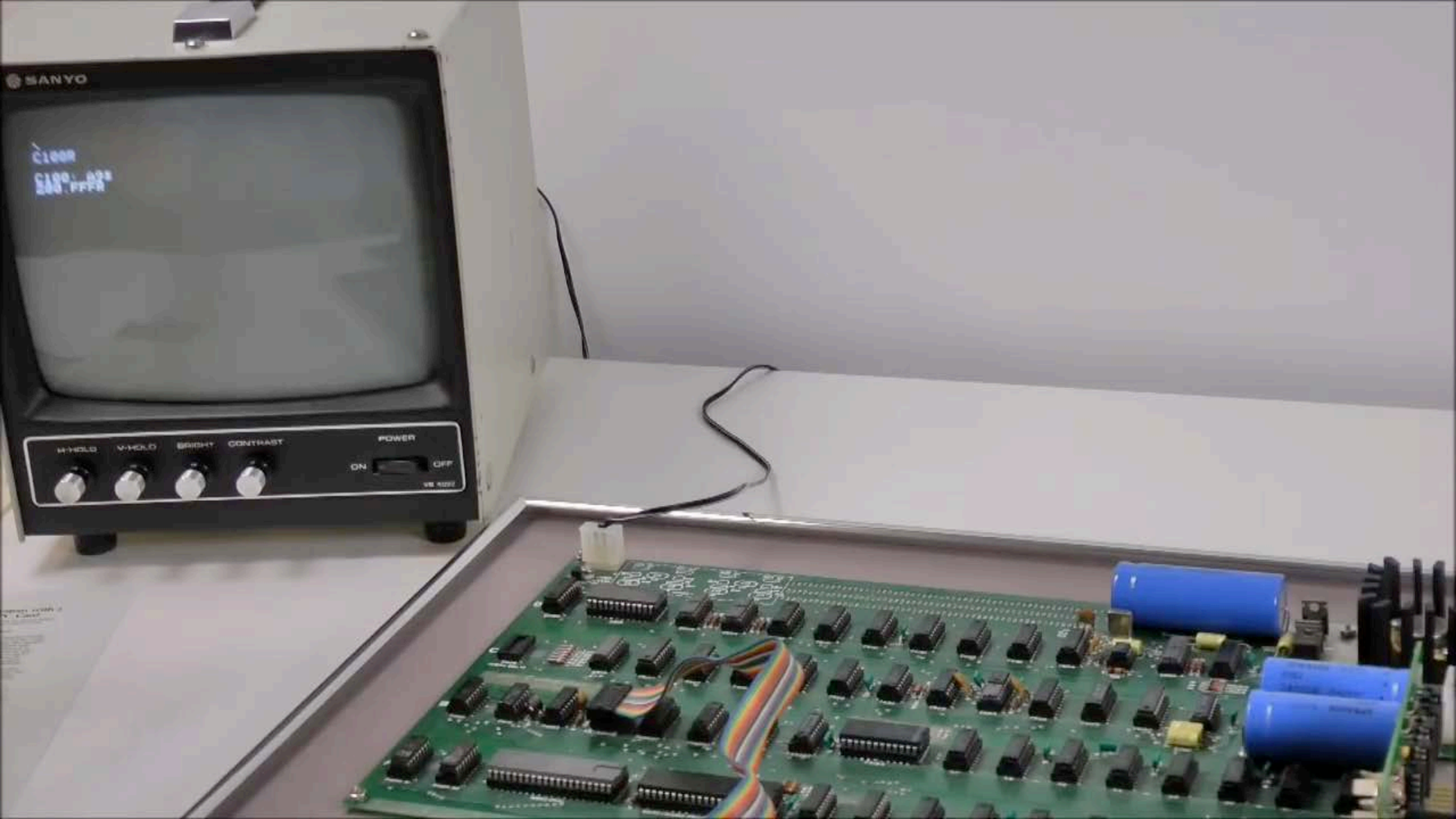
SIZE D | DRAWING NO 00101 | REV A
SCALE | SHEET 2 OF 3

00101 DRAWING NO. | SHEET 2/3 | REV. A



Block diagram of the Apple 1 courtesy of sbproducts.com

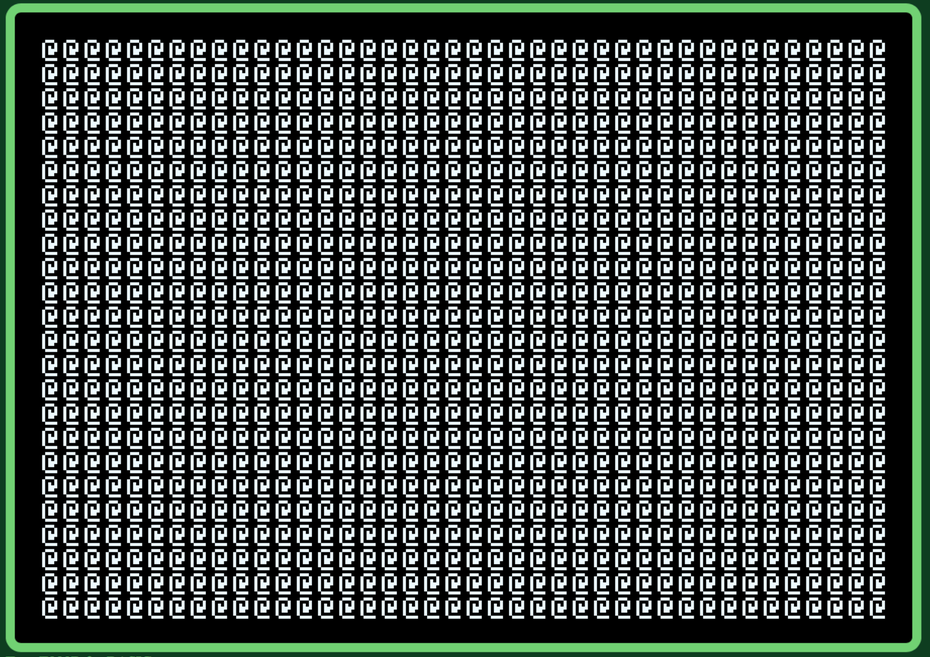

Grafics Demo
LOAD: 280.FFFR
RUN: 280R
apple computer inc.
Graphics Demo


SANYO
H-HOLD
V-HOLD
BRIGHT
CONTRAST
POWER
ON
OFF

www.scullinsteel.com/apple1/#Little%20Tower

Apple 1js

An Apple 1 Emulator in JavaScript



Type E000R for BASIC

1022KHz | Pause | Load | About | Options

Introducing Apple II.™

The home computer that's ready to work, play and grow with you.

Clear the kitchen table. Bring in the color T.V. Plug in your new Apple II,* and connect any standard cassette recorder/player. Now you're ready for an evening of discovery in the new world of personal computers.

Only Apple II makes it that easy. It's a complete, ready to use computer—not a kit. At $1298, it includes features you won't find on other personal computers costing twice as much.

Features such as video graphics in 15 colors. And a built-in memory capacity of 8K bytes ROM and 4K bytes RAM—with room for lots more. But you don't even need to know a RAM from a ROM to use and enjoy Apple II. It's the first personal computer with a fast version of BASIC—the English-like programming language—permanently built in. That means you can begin running your Apple II the first evening, entering your own instructions and watching them work, even if you've had no previous computer experience.

The familiar typewriter-style keyboard makes communication easy. And your programs and data can be stored on (and retrieved from) audio cassettes, using the built-in cassette interface, so you can swap with other Apple II users. This and other peripherals—optional equipment on most personal computers, at hundreds of dollars extra cost—are *built into* Apple II. And it's designed to keep up with changing technology, to expand easily whenever you need it to.

As an educational tool, Apple II is a sound investment. You can program it to tutor your children in most any subject, such as spelling, history or math. But the biggest benefit—no matter *how* you use Apple II—is that you and your family increase your familiarity with the computer itself. The more you experiment with it, the more you discover about its potential.

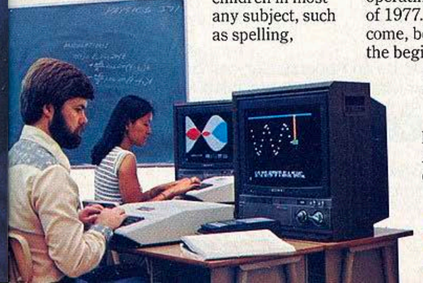
Start by playing PONG. Then invent your own games using the input keyboard, game paddles and built-in speaker. As you experiment you'll acquire new programming skills which will open up new ways to use your Apple II. You'll learn to "paint" dazzling color displays using the unique color graphics commands in Apple BASIC, and write programs to create beautiful kaleidoscopic designs.

As you master Apple BASIC, you'll be able to organize, index and store data on household finances, income tax, recipes, and record collections. You can learn to chart your biorhythms, balance your checking account, even control your home environment. Apple II will go as far as your imagination can take it.

Best of all, Apple II is designed to grow with you. As your skill and experience with computing increase, you may want to add new Apple peripherals. For example, a refined, more sophisticated BASIC language is being developed for advanced scientific and mathematical applications.

And in addition to the built-in audio, video and game interfaces, there's room for eight plug-in options such as a prototyping board for experimenting with interfaces to other equipment; a serial board for connecting teletype, printer and other terminals; a parallel interface for communicating with a printer or another computer; an EPROM board for storing programs permanently; and a modem board communications interface. A floppy disk interface with software and complete operating systems will be available at the end of 1977. And there are many more options to come, because Apple II was designed from the beginning to accommodate increased power and capability as your requirements change.

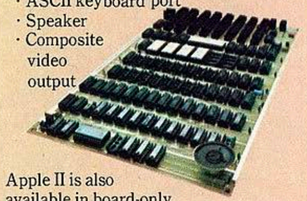
If you'd like to see for yourself how easy it is to use and enjoy Apple II, visit your local dealer for a demonstration and a copy of our detailed brochure. Or write Apple Computer Inc., 20863 Stevens Creek Blvd., Cupertino, California 95014.



Apple II™ is a completely self-contained computer system with BASIC in ROM, color graphics, ASCII keyboard, lightweight, efficient switching power supply and molded case. It is supplied with BASIC in ROM, up to 48K bytes of RAM, and with cassette tape, video and game I/O interfaces built-in. Also included are two game paddles and a demonstration cassette.

SPECIFICATIONS

- **Microprocessor:** 6502 (1 MHz).
- **Video Display:** Memory mapped, 5 modes—all Software-selectable:
  - Text—40 characters/line, 24 lines upper case.
  - Color graphics—40h x 48v, 15 colors
  - High-resolution graphics—280h x 192v; black, white, violet, green (16K RAM minimum required)
  - Both graphics modes can be selected to include 4 lines of text at the bottom of the display area.
  - Completely transparent memory access. All color generation done digitally.
- **Memory:** up to 48K bytes on-board RAM (4K supplied)
  - Uses either 4K or new 16K dynamic memory chips
  - Up to 12K ROM (8K supplied)
- **Software**
  - Fast extended Integer BASIC in ROM with color graphics commands
  - Extensive monitor in ROM
- **I/O**
  - 1500 bps cassette interface
  - 8-slot motherboard
  - Apple game I/O connector
  - ASCII keyboard port
  - Speaker
  - Composite video output



Apple II is also available in board-only form for the do-it-yourself hobbyist. Has all of the features of the Apple II system, but does not include case, keyboard, power supply or game paddles. $798.

PONG is a trademark of Atari Inc.

*Apple II plugs into any standard TV using an inexpensive modulator (not supplied).

Circle 4 on inquiry card.

apple computer inc.™

Circle 4 on inquiry card.

The home computer that's ready to work, play and grow with you.

Clear the kitchen table. Bring in the color T.V. Plug in your new Apple II,* and connect any standard cassette recorder/player. Now you're ready for an evening of discovery in the new world of personal computers.

Only Apple II makes it that easy. It's a complete, ready to use computer—not a kit. At $1298, it includes features you won't find on other personal computers costing twice as much.

history or math. But the biggest benefit—no matter *how* you use Apple II—is that you and your family increase your familiarity with the computer itself. The more you experiment with it, the more you discover about its potential.

Start by playing PONG. Then invent your own games using the input keyboard, game paddles and built-in speaker. As you experiment you'll acquire new programming skills which will open up new ways to use your Apple II. You'll learn to "paint" dazzling color displays using the unique color graphics commands in Apple BASIC, and write programs to create beautiful kaleidoscopic designs.

As you master Apple BASIC, you'll be able to organize, index and store data on household finances, income tax, recipes, and record collections. You can learn to chart your biorhythms,

Apple II™ is a completely self-contained computer system with BASIC in ROM, color graphics, ASCII keyboard, lightweight, efficient switching power supply and molded case. It is supplied with BASIC in ROM, up to 48K bytes of RAM, and with cassette tape, video and game I/O interfaces built-in. Also included are two game paddles and a demonstration cassette.

SPECIFICATIONS

- **Microprocessor:** 6502 (1 MHz).
- **Video Display:** Memory mapped, 5 modes—all Software-selectable:
  - Text—40 characters/line, 24 lines upper case.
  - Color graphics—40h x 48v, 15 colors
  - High-resolution graphics—280h x 192v; black, white, violet, green (16K RAM minimum required)
  - Both graphics modes can be selected


drive 1
disk II
drive 2
disk II
apple][

Apple][Boot Process

1. Power on
2. Loads initial code from ROM to initialise the hardware
3. Load Integer BASCI (or System Monitor) from ROM
4. User initiates boot from expansion code ROM (here slot 6) using
   `PR#6` (Integer BASIC) or
   `C600G` (System Monitor)

The Apple II Plus included the ability to scan each expansion slot for a bootable expansion card ROM and automatically call it (starting with slot 7 down to slot 1).

Apple][Boot Process

5. Expansion card ROM boot code attempts to boot from drive 1 of the controller by reading 256 bytes from sector 0 of track 0.
6. Sector zero contains a program to read sectors 0 through 9 of track 0 into memory using part of the ROM boot code.
7. The program in sectors 1-9 of track 0, including the complete RWTS code, proceeds to load tracks 1 and 2 – the rest of DOS.
8. Relocate DOS as high into system memory as possible (System master disks determine the computer's RAM config).
9. Once DOS is loaded into memory, it attempts to load and execute a startup program as indicated in the DOS program code.

@CHM
Overview

MUSEUM BLOG
Overview
Categories
Authors
Archives

PODCASTS
CHM Live
From The Archives

Home

Apple II DOS Source Code

Len Shustek Nov 12, 2013 From the Collection

Software Gems: The Computer History Museum Historical Source Code Series

Could you write a Disk Operating System in 7 weeks?

In June 1977 Apple Computer shipped their first mass-market computer: the Apple II.

Introducing Macintosh. What makes it tick. And talk.

Well, to begin with, 110 volts of alternating current.

Secondly, some of the hottest hardware to come down the pike in the last 3 years.

The garden variety 16-bit 8088 microprocessor.

Macintosh's 32-bit MC68000 microprocessor.

Some hard facts may be in order at this point:

Macintosh's brain is the same blindingly-fast 32-bit microprocessor we gave our other brainchild, the Lisa™ Personal Computer. Far more powerful than the 16-bit 8088 found in current generation computers.

Its heart is the same Lisa Technology of windows, pull-down menus, mouse commands and icons. All of which make that 32-bit power far more useful by making the Macintosh™ Personal Computer far easier to use than current generation computers. In fact, if you can point without hurting yourself, you can use it.

Now for some small talk.

Thanks to its size, if you can't bring the problem to a Macintosh, you can always bring a Macintosh to the problem. (It weighs 9 pounds less than the most popular "portable.")

Another miracle of miniaturization is Macintosh's built-in 3½" drive. Its disks store 400K—more than conventional 5¼" floppies. So while they're big enough to hold a desk full of work, they're small enough to fit in a shirt pocket. And, they're totally encased in a rigid plastic so they're totally protected.

And talk about programming.

There are already plenty of programs to keep a Macintosh busy. Like MacPaint,™

a program that, for the first time, lets a personal computer produce virtually any image the human hand can create. There's more software on the way from developers like Microsoft,® Lotus,™ and Software Publishing Corp., to mention a few.

Macintosh automatically makes room for your illustrations in the text.

MacPaint produces virtually any image the human hand can create.

Microsoft's Multiplan for Macintosh.

And with Macintosh BASIC, Macintosh Pascal and our Macintosh Toolbox for writing your own mouse-driven programs, you, too, could make big bucks in your spare time.

You can even program Macintosh to talk in other languages, like Yiddish or Serbo-Croation, because it has a built-in polyphonic sound generator capable of producing high quality speech or music.

The Mouse itself. Replaces typed-in computer commands with a form of communication you already understand — pointing.

Some mice have two buttons. Macintosh has one. So it's extremely difficult to push the wrong button.

The inside story — a rotating ball and optical sensors translate movements of the mouse to Macintosh's screen pointer with pin-point accuracy.

All the right connections.

On the back of the machine, you'll find built-in RS232 and RS422 AppleBus serial communication ports. Which means you can connect printers, modems and other peripherals without adding $150 cards. It also means that Macintosh is ready to hook in to a local area network. (With AppleBus, you will be able to interconnect up to 16 different Apple computers and peripherals.)

Should you wish to double Macintosh's storage with an external disk drive, you can do so without paying for a disk controller card—that connector's built-in, too.

There's also a built-in connector for Macintosh's mouse, a feature that costs up to $300 on computers that can't even run mouse-controlled software.

One last pointer.

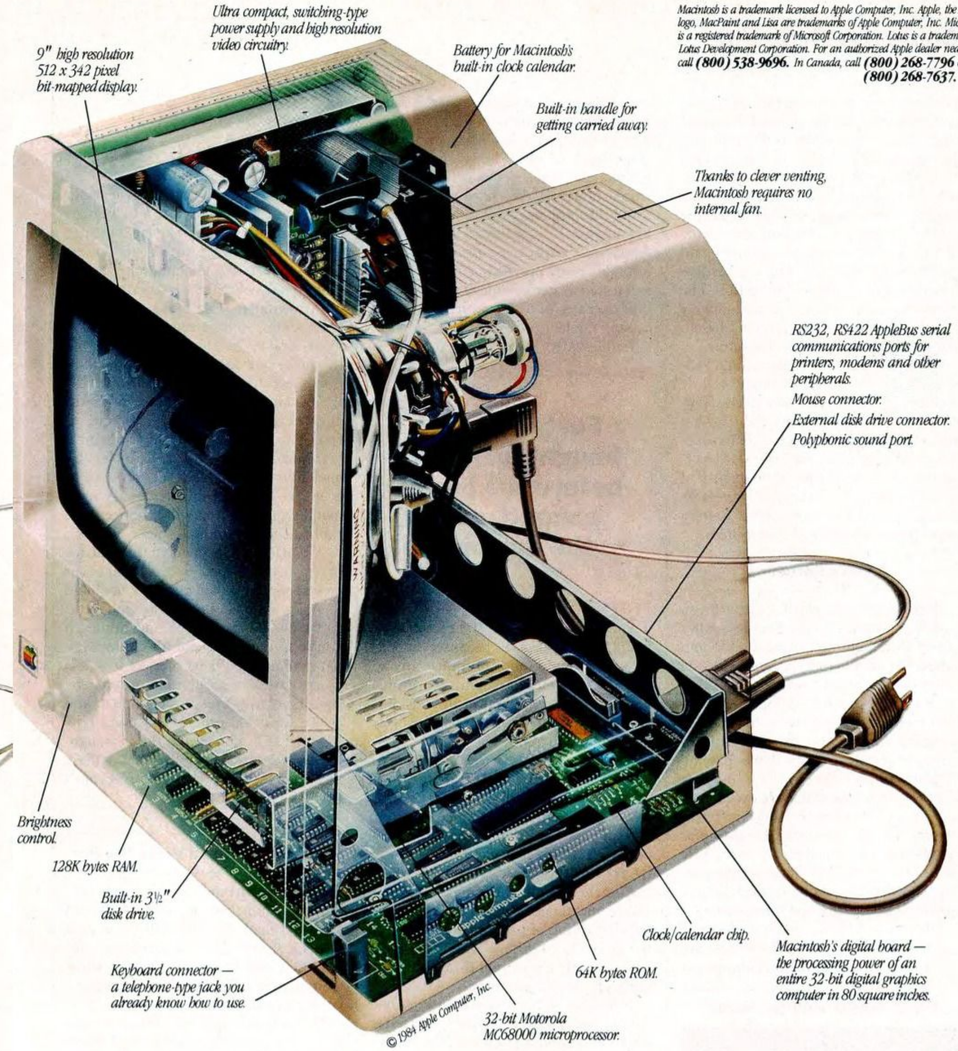
Now that you've seen some of the logic, the technology, the engineering genius and the software wizardry that separates Macintosh from conventional computers, we'd like to point you in the direction of your nearest authorized Apple dealer.

Over 1500 of them are eagerly waiting to put a mouse in your hand. As one point-and-click makes perfectly clear, the real genius of Macintosh isn't its 32-bit Lisa Technology, or its 3½" floppy disks, or its serial ports, or its software, or its polyphonic sound generator.

The real genius is that you don't have to be a genius to use a Macintosh.

You just have to be smart enough to buy one.

Soon there'll be just two kinds of people. Those who use computers. And those who use Apples.



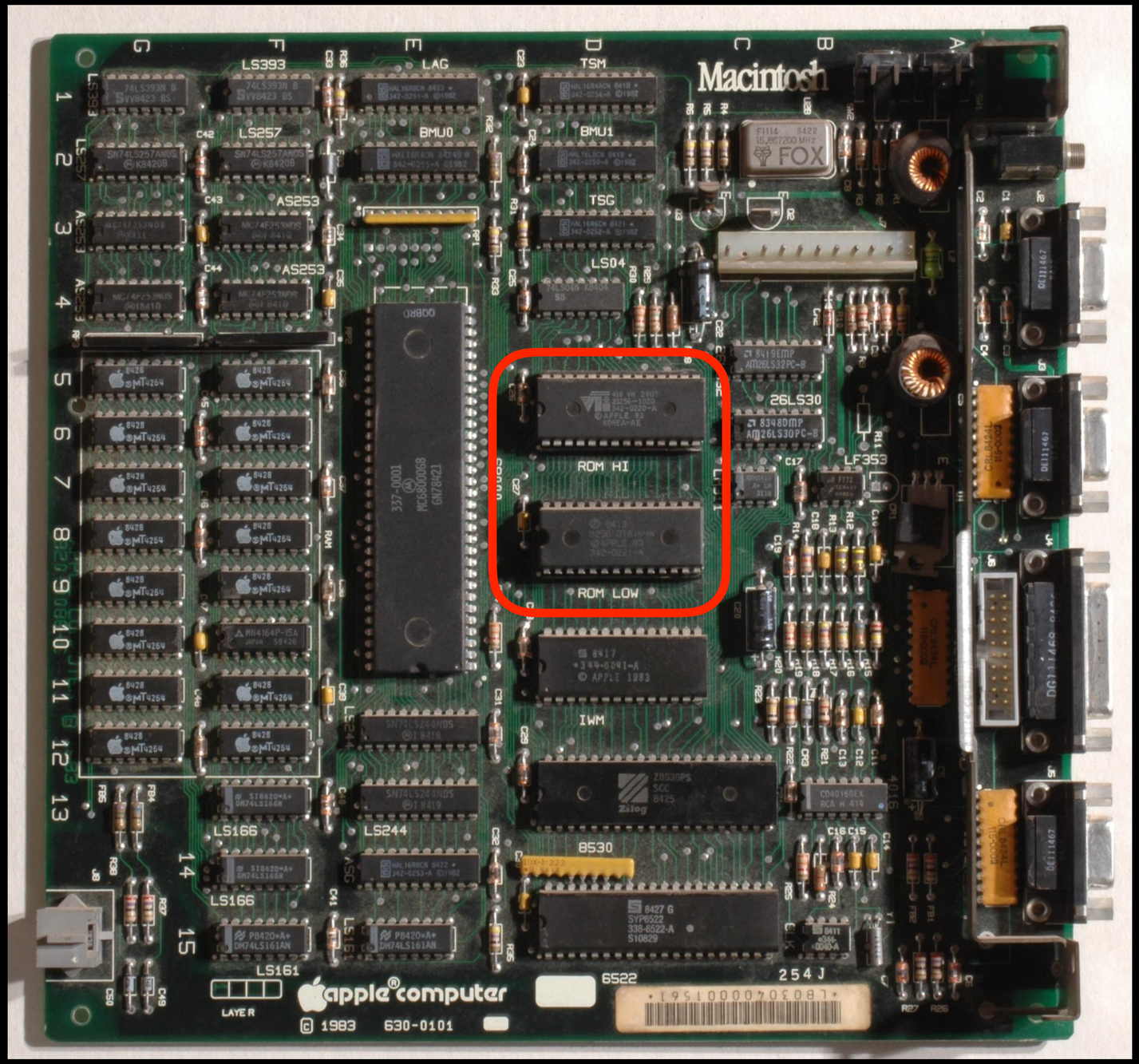


Photo: Shieldforyoureyes Dave Fischer

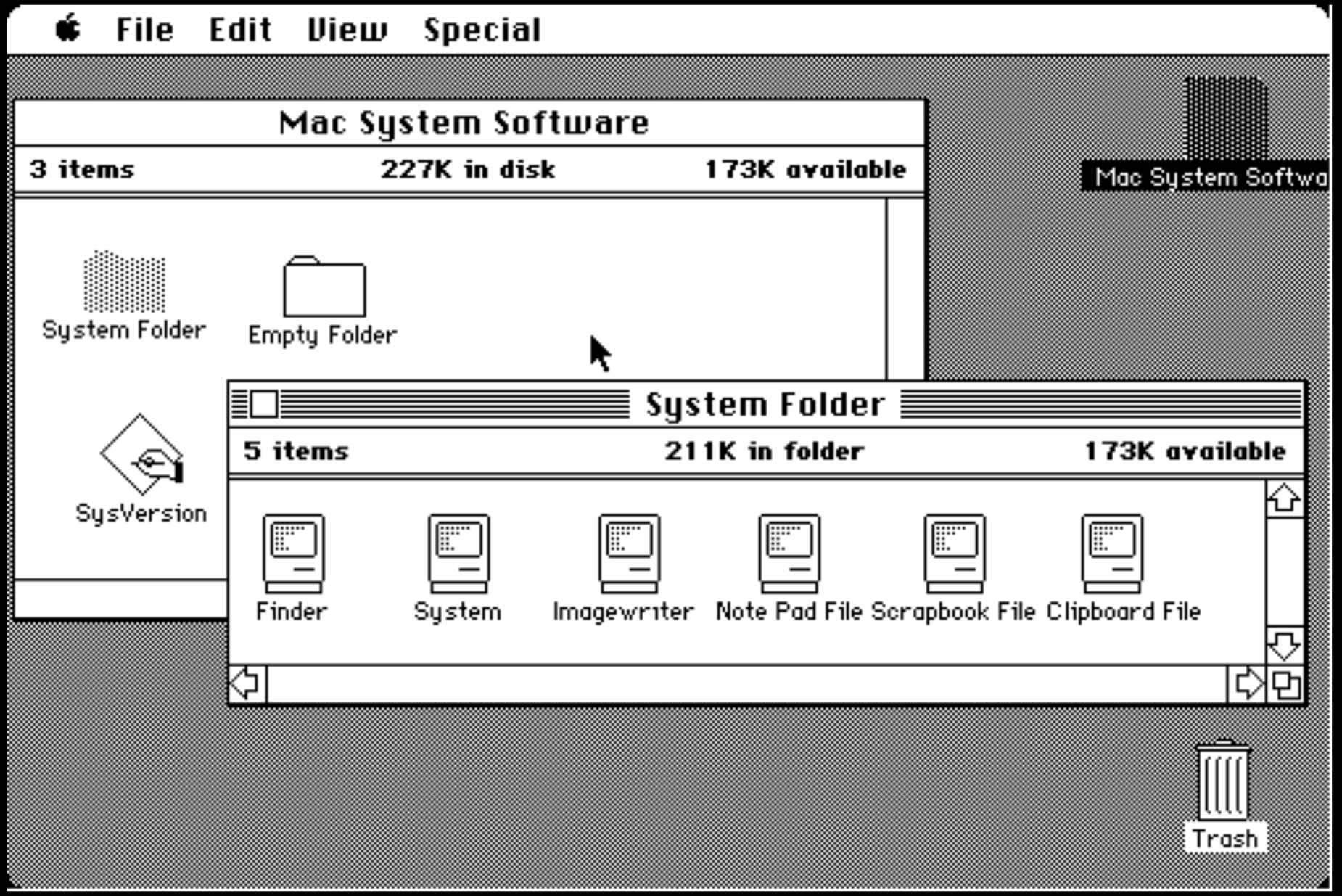
Apple Macintosh Boot Process

1. Power on
2. Load Macintosh Toolbox from ROM
3. Toolbox executes memory test
4. Toolbox enumerates devices
5. Toolbox start on-board video
   or optional ROM from NuBus or PCI video card
6. Toolbox check for floppy disk
   and scan all SCSI buses for disks with a valid System Folder

Apple Macintosh Boot Process

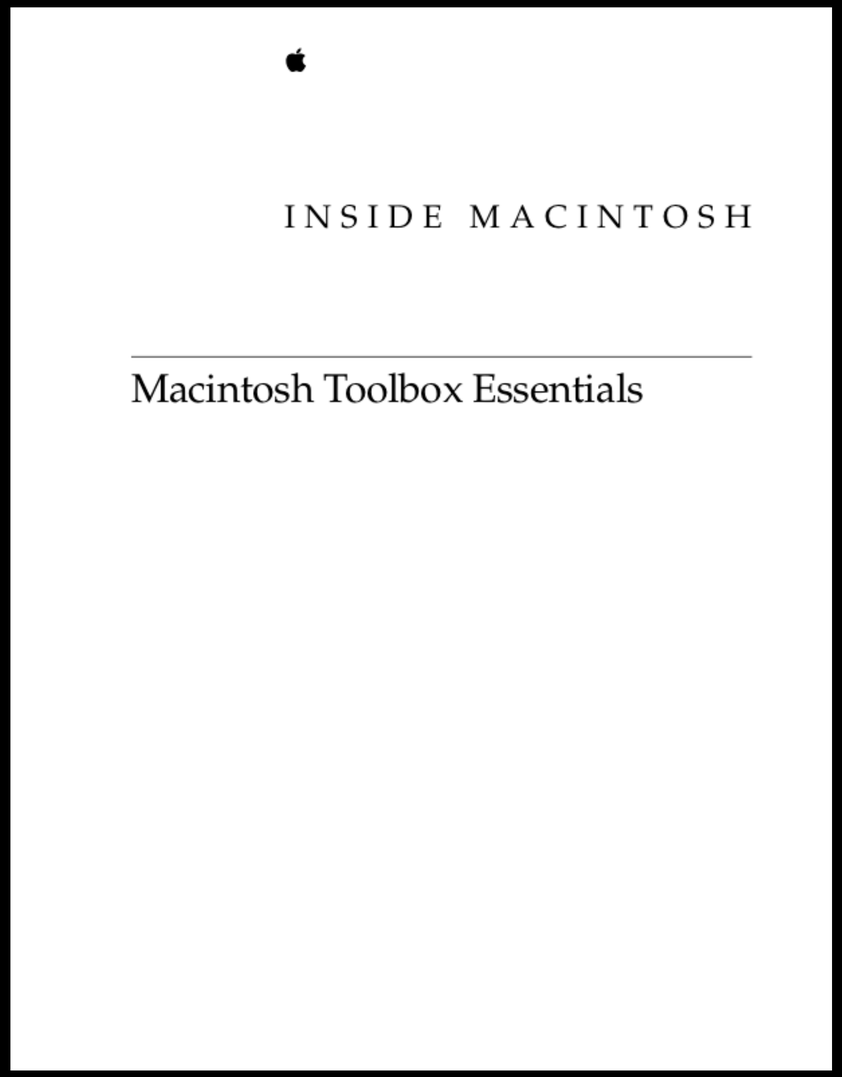
7. Attempt to start the system
   giving preference to setting in parameter RAM:
   a. Happy Mac logo is displayed, if bootable disk is found.
      Hand over control to Mac OS.
   b. Floppy disk with flashing question mark is displayed,
      if no disk to boot from is present.
   c. Sad Mac icon and hexadecimal error code are displayed,
      if a hardware problem occurs during the early part of the boot process.
      This will be accompanied by the Chimes of Death on Macs made after 1987.

0 0 0 0 0 0 0 F
0 0 0 0 0 0 0 3



Original Macintosh System Software 1.0

Macintosh Toolbox



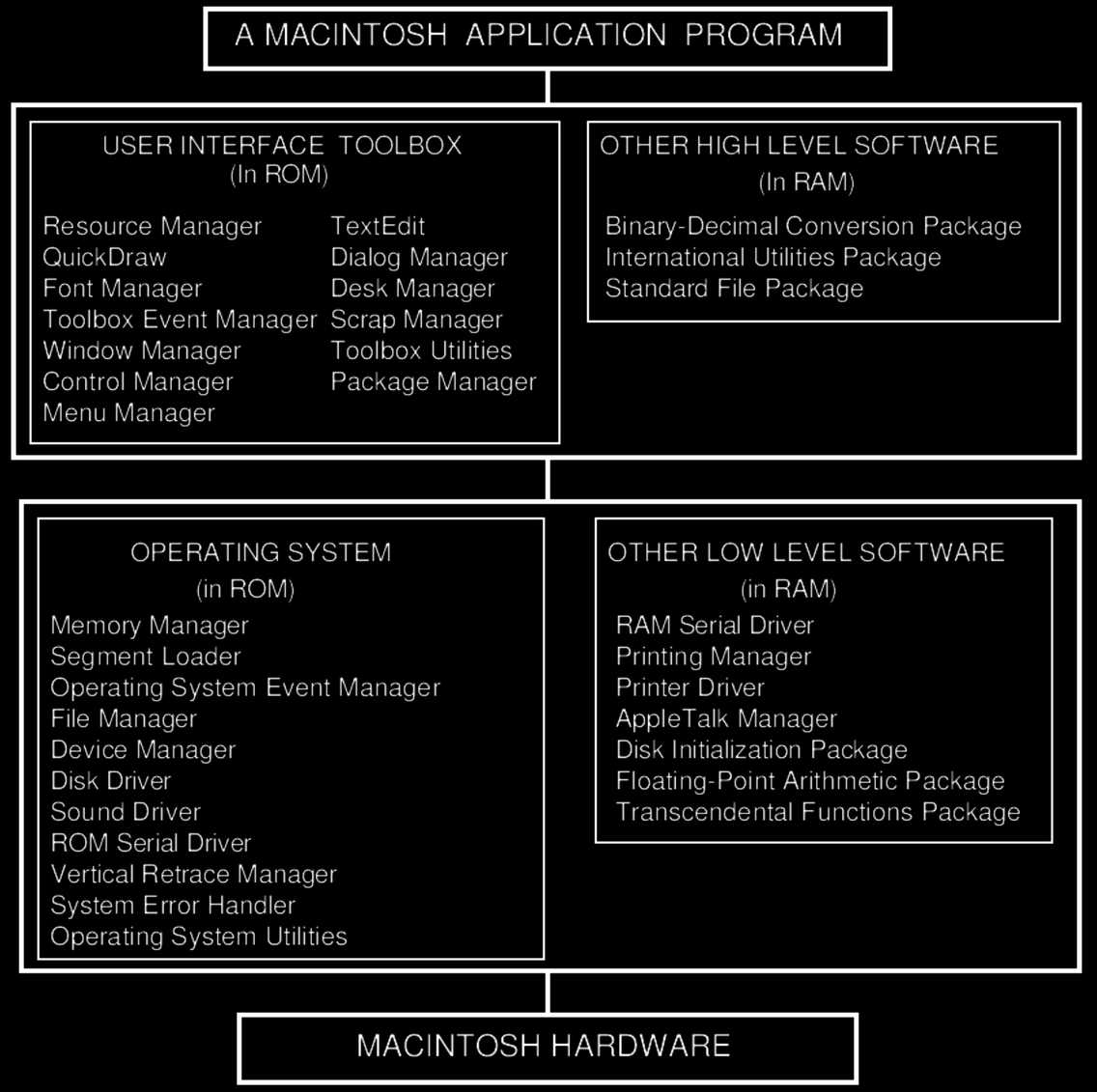
"The Macintosh Toolbox encompasses a number of system software routines, most (but not all) of which help present your application's interface to the user.

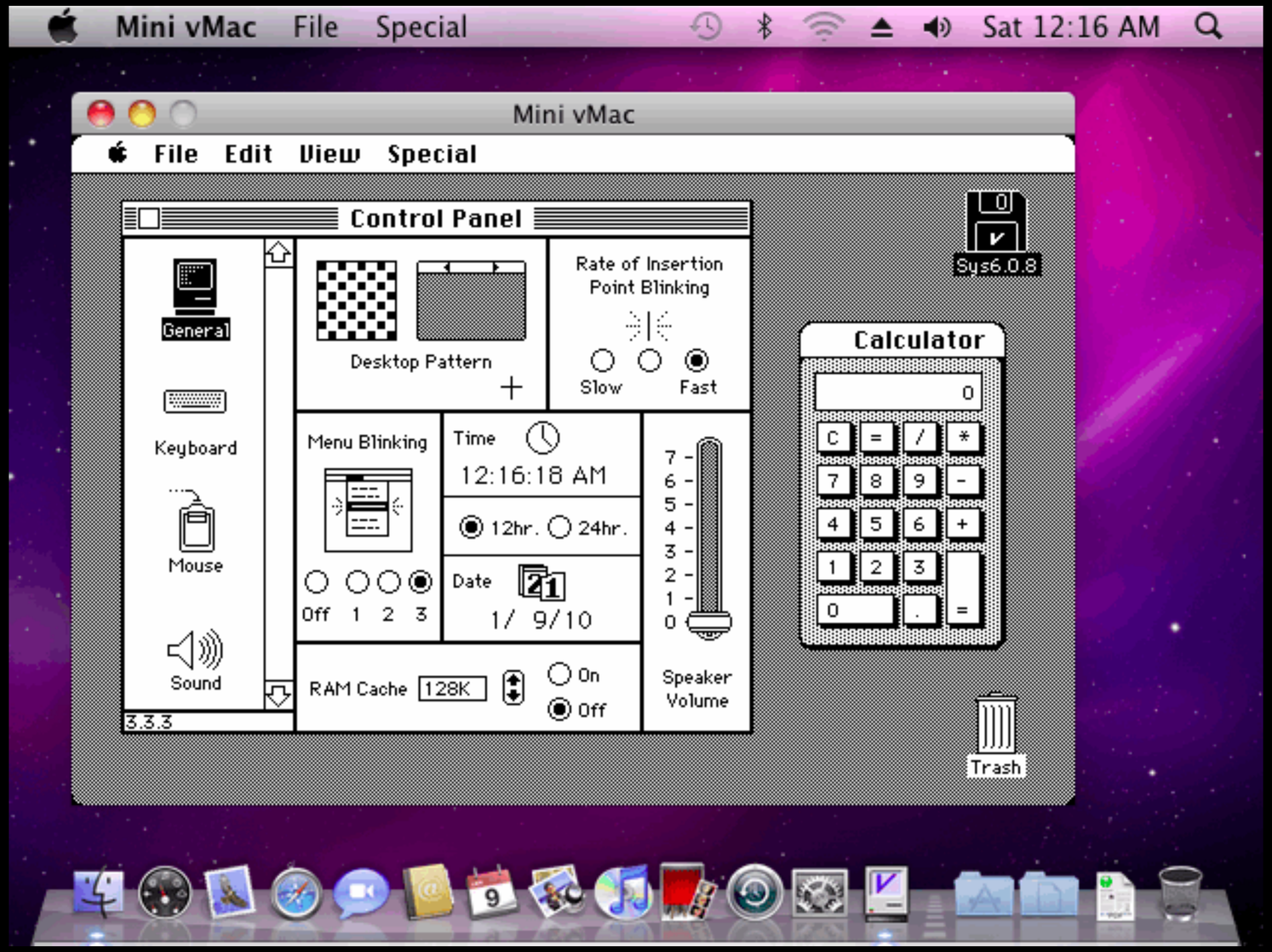
Some of these routines include those provided by the Event Manager, Menu Manager, Window Manager, Control Manager, Dialog Manager, Help Manager, Resource Manager, and Scrap Manager."

https://developer.apple.com/legacy/library/documentation/mac/pdf/MacintoshToolboxEssentials.pdf

Macintosh Toolbox

- 64kB ROM
  - A-trapping: clever usage of the M68k's illegal opcode exception handling, but required emulation on PowerPc
  - ROM increasingly hard to maintain
- Low level code:
  - hardware initialisation & drivers
  - diagnostics
- High-level code:
  - User interface components
  - Operating system components
- Most of the Toolbox moved to Carbon API





https://www.gryphel.com/c/minivmac/screens/index.html


hello
(again)
iMac

Technical Note TN1167

The Mac ROM Enters a NewWorld

CONTENTS

The Mac ROM is "different" starting with the iMac. Come along and find out what's new.

This Technote describes changes made to the Macintosh ROM since the introduction of the iMac.

The Macintosh ROM, sometimes called the ToolBox ROM, has been updated. The ToolBox (including the OS) has been removed from the ROM; the ROM physical size, Macintosh memory map, and boot sequence have also been changed.

This Technote describes the changes to the new Apple ROM called NewWorld, which will be the ROM used on all future Macintoshes. This Note is directed at device developers who have devices such as PCI, USB, and FireWire (especially device types that could participate in the boot sequence).

Updated: [May 17 1999]

Introduction

The NewWorld Architecture is the basis for Mac OS startup and ToolBox functionality for all Macintosh CPUs beginning with iMac. This document is designed to help developers understand the organization of the NewWorld Architecture and some ways to use it to best advantage.

This document describes how the NewWorld Architecture works from an organizational and execution flow standpoint, and describes differences from older architectures. It briefly covers the "Old World" ROM organization as background, then explains the NewWorld Architecture and execution flow.

Familiarity with the traditional Macintosh ROM structure is useful when reading this document.

While the focus of this document is on Mac OS, the Boot ROM and "bootinfo" components of the NewWorld Architecture are designed to be operating-system independent. Furthermore, the mechanisms behind the engineering techniques used for Mac OS can be applied to other operating systems.

https://web.archive.org/web/20010413153004/http://developer.apple.com:80/technotes/tn/tn1167.html

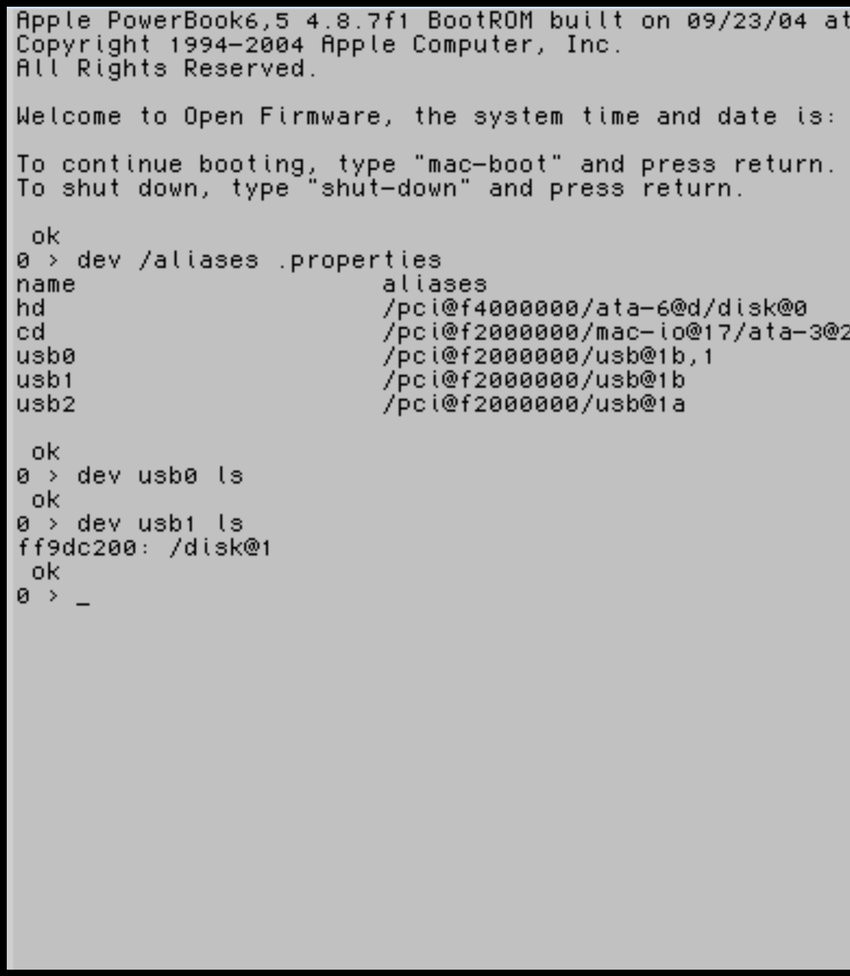
New World Boot Architecture

1. Boot ROM – the hardware-specific firmware
   - power-on self test (POST) and diagnostics
   - start-up code without Mac OS-specific code
   - Open Firmware (version 3.0)
   - Mac OS drivers for motherboard devices needed at boot time (ndrv's and nlib's)

2. `bootinfo` file is kept in the system folder of the startup volume
   - Mac OS-specific Open Firmware code and required "bootinfo" components
   - Open Firmware-specific Mac OS code ("Trampoline" code)
   - Mac OS Toolbox ROM Image (`Mac OS ROM` file in system folder)
   - and other Mac OS software

Open Firmware / OpenBoot

- IEEE 1275-1994 standard
  originated at Sun, also used by Apple, IBM, and ARM
- Forth language based shell
  Allows to interactively develop and test
- FCode is highly compact platform independent byte code
  Allows for platform-independent boot-time diagnostics, configuration code, and device drivers on PCI cards.
- Standardised system hardware description
  Reducing need for user configuration and hardware polling.
- PowerPC Macs: ⌘ Cmd + ⌥ Option + O + F



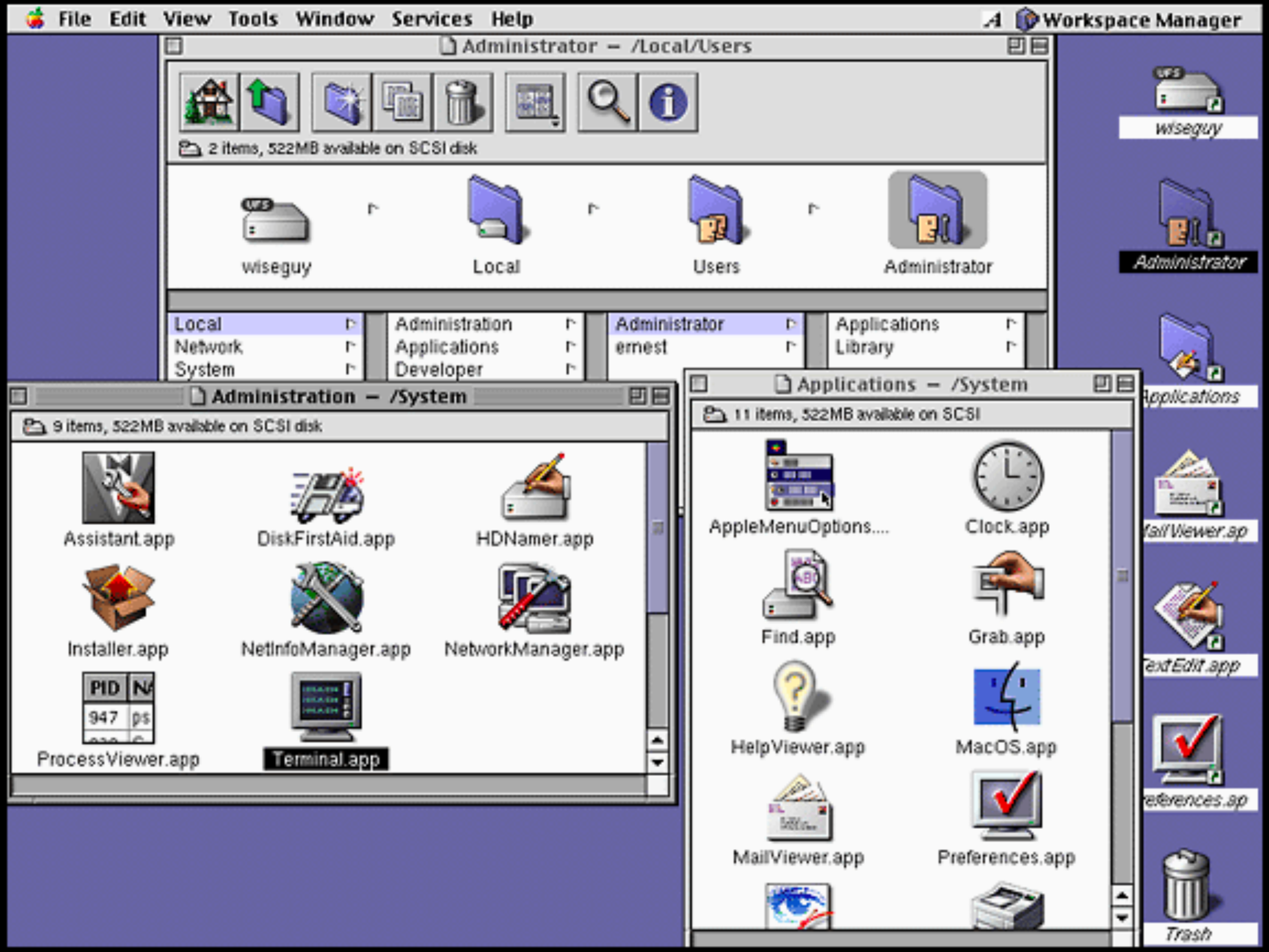
New World Boot Process Overview

1. User presses power key.
2. Between the time that the power key is pressed and the boot beep is heard, while the screen is still black, a ROM checksum is taken, the processor is checked, the interrupt controller is started, all the clocks are determined, the memory controller is initialized, NVRAM is checked, RAM is sized checked and initialized, and the L2 cache is sized and prepared (L2 cache is enabled in POST).
3. The POST code runs: preliminary diagnostics, boot beep, initialisation, and setup.
4. Open Firmware initialises and begins execution, including building the Device Tree.

New World Boot Process Overview

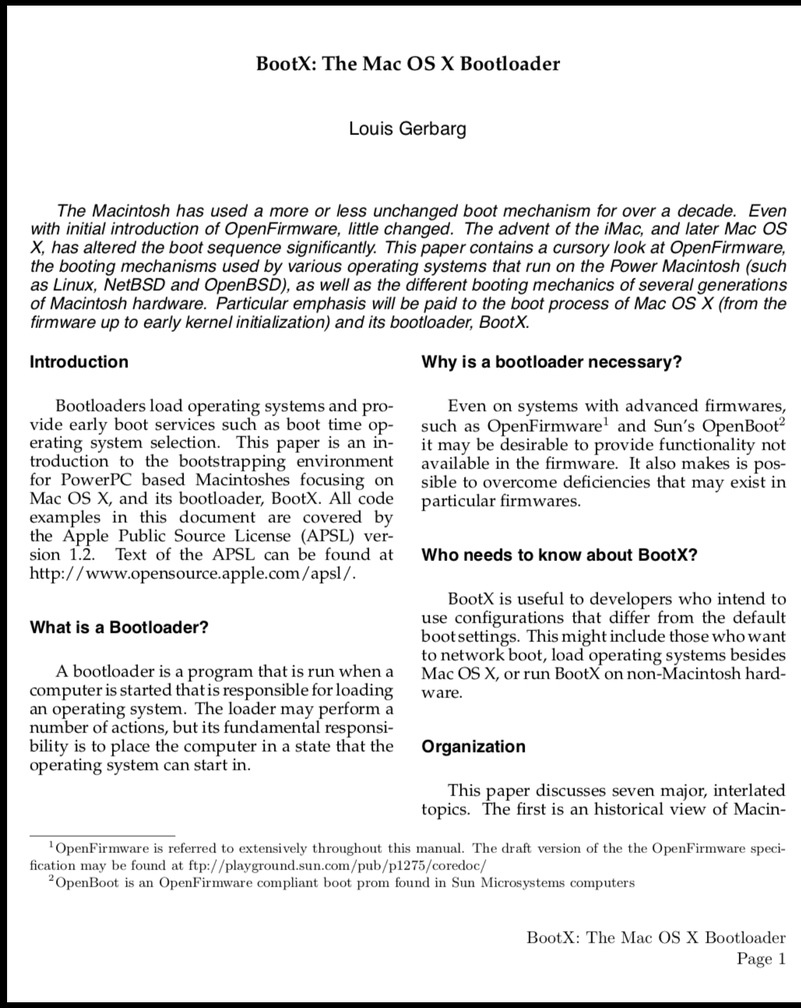
5. Open Firmware looks for a boot device and loads the `bootinfo` file (based on defaults and NVRAM settings).
6. Open Firmware executes the `bootinfo` Forth script to read the Trampoline code and the Toolbox ROM Image (`'Mac OS ROM'` image file of type tbxi in system folder).
7. `bootinfo` script transfers control to the Trampoline code, which functions as the transition between Open Firmware and the beginning of the Mac OS execution.
8. The Trampoline code gathers information about the system from Open Firmware, creates data structures based on this information, and terminates Open Firmware.
9. The Trampoline code transfers control to the Toolbox ROM Image initialisation code.


Grab File Edit Capture Window Help
Mon 4:18 PM
Macintosh HD
About This Mac
Mac OS X
(Build 4K78)
Memory: 320 MB
Processor: PowerPC G4
™ & © Apple Computer. Inc. 1983-2001


File Edit View Tools Window Services Help
Workspace Manager
Administrator — /Local/Users
2 items, 522MB available on SCSI disk
wiseguy
Local
Users
Administrator
Local
Network
System
Administration
Applications
Developer
Administrator
ernest
Applications
Library
Administration — /System
9 items, 522MB available on SCSI disk
Assistant.app
DiskFirstAid.app
HDNamer.app
Installer.app
NetInfoManager.app
NetworkManager.app
ProcessViewer.app
Terminal.app
Applications — /System
11 items, 522MB available on SCSI
AppleMenuOptions....
Clock.app
Find.app
Grab.app
HelpViewer.app
MacOS.app
MailViewer.app
Preferences.app
wiseguy
Administrator
Trash

BootX

- BootX is a boot loader for use on the PCI Power Macintosh to start-up Mac OS X.
- BootX can load kernels in Mach-o and ELF formats. This allows it to load kernels or second stage loaders for Mac OS X, Linux, OpenBSD, NetBSD, and FreeBSD.
- BootX understands a diverse series of filesystems: HFS+, 4.4 BSD Big Endian UFS (Mac OS X only, slightly incompatible with the other BSDs), ext2, as well as loading kernels over any OpenFirmware via tftp.
- The program is freely available as part of the Darwin OS under the Apple Public Source License.



BootX: The Mac OS X Bootloader

Louis Gerbarg

The Macintosh has used a more or less unchanged boot mechanism for over a decade. Even with initial introduction of OpenFirmware, little changed. The advent of the iMac, and later Mac OS X, has altered the boot sequence significantly. This paper contains a cursory look at OpenFirmware, the booting mechanisms used by various operating systems that run on the Power Macintosh (such as Linux, NetBSD and OpenBSD), as well as the different booting mechanics of several generations of Macintosh hardware. Particular emphasis will be paid to the boot process of Mac OS X (from the firmware up to early kernel initialization) and its bootloader, BootX.

Introduction

Bootloaders load operating systems and provide early boot services such as boot time operating system selection. This paper is an introduction to the bootstrapping environment for PowerPC based Macintoshes focusing on Mac OS X, and its bootloader, BootX. All code examples in this document are covered by the Apple Public Source License (APSL) version 1.2. Text of the APSL can be found at http://www.opensource.apple.com/apsl/.

What is a Bootloader?

A bootloader is a program that is run when a computer is started that is responsible for loading an operating system. The loader may perform a number of actions, but its fundamental responsibility is to place the computer in a state that the operating system can start in.

Why is a bootloader necessary?

Even on systems with advanced firmwares, such as OpenFirmware[1] and Sun's OpenBoot[2] it may be desirable to provide functionality not available in the firmware. It also makes is possible to overcome deficiencies that may exist in particular firmwares.

Who needs to know about BootX?

BootX is useful to developers who intend to use configurations that differ from the default boot settings. This might include those who want to network boot, load operating systems besides Mac OS X, or run BootX on non-Macintosh hardware.

Organization

This paper discusses seven major, interlated topics. The first is an historical view of Macin-

[1] OpenFirmware is referred to extensively throughout this manual. The draft version of the the OpenFirmware specification may be found at ftp://playground.sun.com/pub/p1275/coredoc/

[2] OpenBoot is an OpenFirmware compliant boot prom found in Sun Microsystems computers

https://web.archive.org/web/20070309142504/http://www.cs.rpi.edu/~gerbal/BootX.pdf

Mac OS X Boot Process on PowerPC

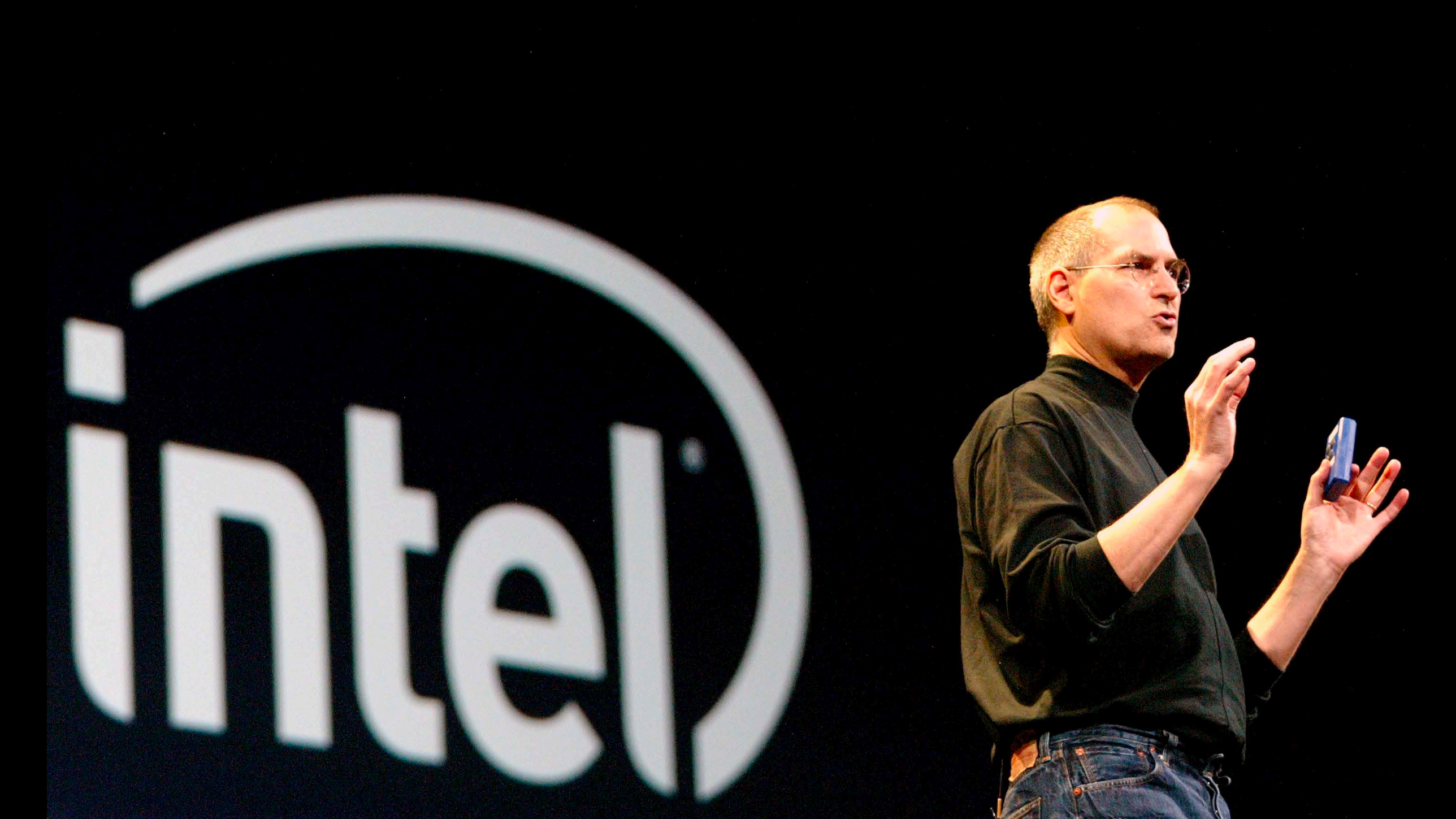
1. User powers on device
2. Machine runs low-level initialisation
3. Open Firmware initialises and begins execution, including building the Device Tree.
4. Open Firmware looks for a boot device and loads the `bootinfo` file (based on defaults and NVRAM settings).
5. Open Firmware executes the `bootinfo` Forth script and loads loads a file `BootX` of type tbxi from the boot device `/System/Library/CoreServices` and executes `BootX`.

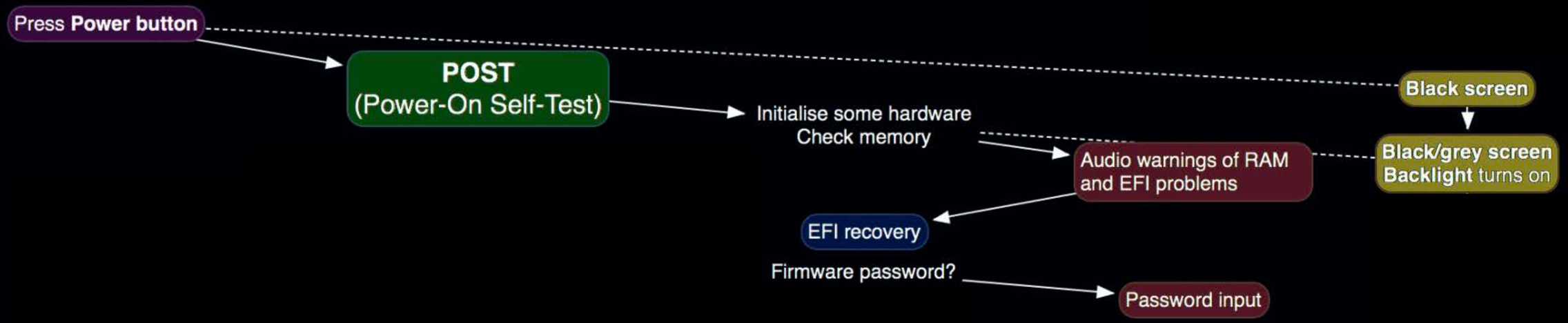
https://web.archive.org/web/20080706145146/http://developer.apple.com/documentation/MacOSX/Conceptual/BPSystemStartup/Articles/BootProcess.html

Mac OS X Boot Process on PowerPC

6. BootX reads root partition out of nvram.
7. BootX loads mach kernel from the device.
8. BootX copies Mac OS X device drivers from partition into memory.
9. BootX disables all address translations.
10. BootX starts Mac OS X mach kernel.
11. mach kernel begins its boot process.
12. mach kernel may use an integrated linker to link Mac OS X device drivers into itself if it is necessary to complete booting.
13. mach kernel unlinks the integrated linker to save memory.
14. mach_init, init, or launchd take over as root system process.

With Mac OS X 10.2 Apple changed from the coloured 'Happy Mac' to the grey Apple logo as boot splash.


intel


Press **Power button**
POST
(Power-On Self-Test)
Initialise some hardware
Check memory
Black screen
Black/grey screen
Backlight turns on
Audio warnings of RAM
and EFI problems
EFI recovery
Firmware password?
Password input

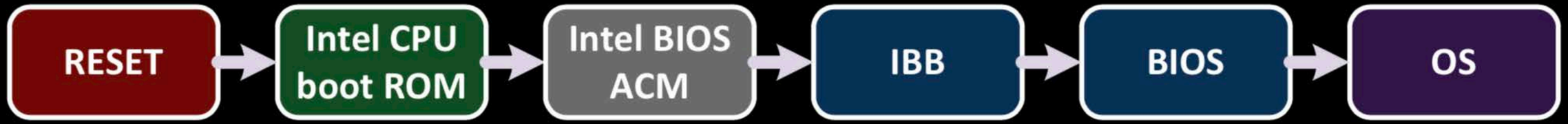
Intel Boot Guard

Intel Boot Guard is a processor feature that prevents the computer from running firmware images not released by the system manufacturer. When turned on, the processors verifies a signature contained in the firmware image before executing it, using the hash of the public half of the signing key, which is fused into the system's Platform Controller Hub (PCH) by the system manufacturer (not by Intel).

Wikipedia: Intel vPro

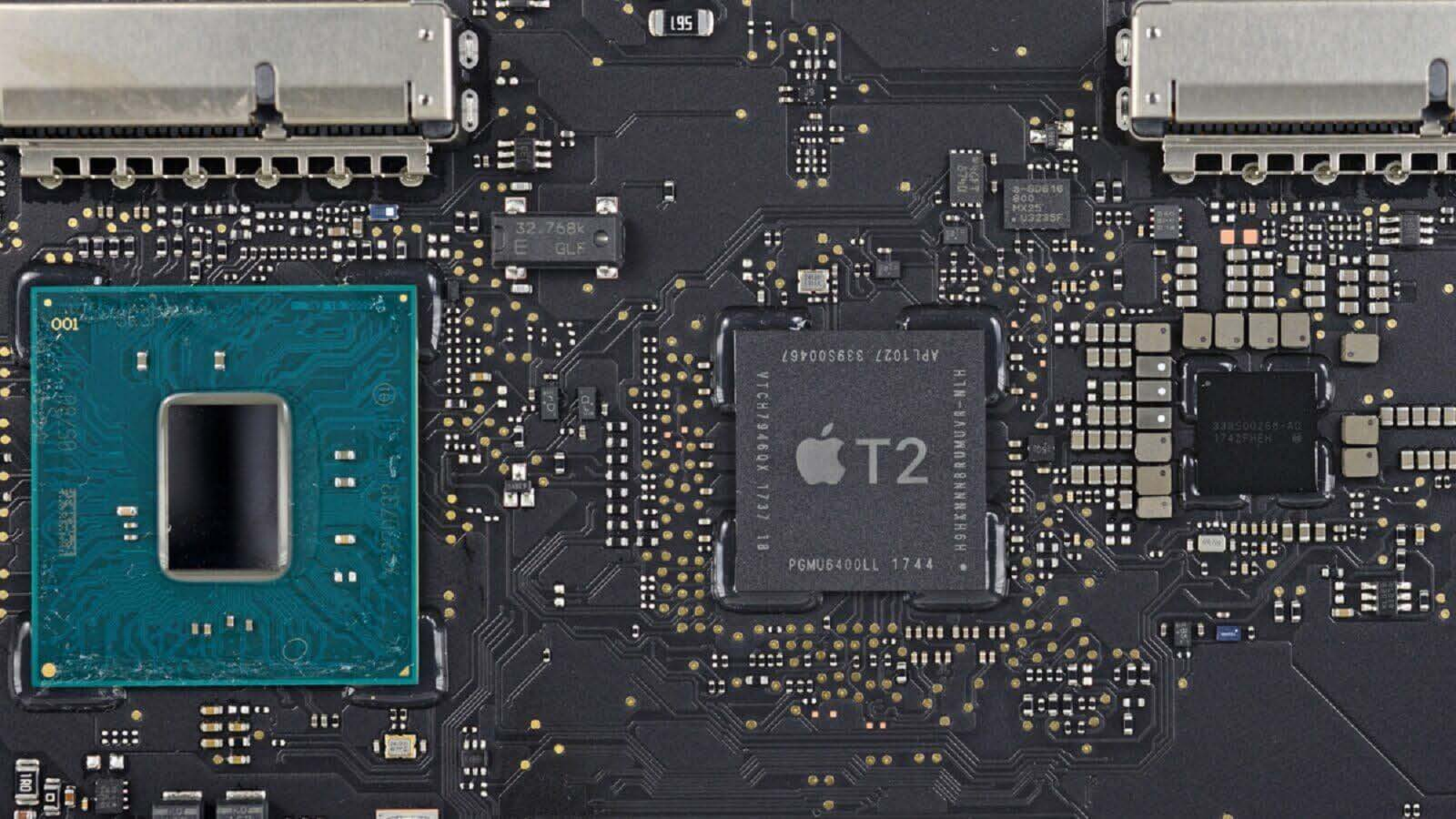
https://en.wikipedia.org/wiki/Intel_vPro#Intel_Boot_Guard

Intel Boot Guard



- Proprietary closed standard implementing a Verified Boot, Measured Boot, and Verified + Measured Boot mode.
- Goal is to cryptographically verify the integrity of the initial boot block (IBB).
  IBB could be seen as all boot components with the CPU doing the initial verifications.
- Root of Trust stored in Field Programmable Fuses (FPFs),
  programmed during manufacturing mode and some of PC vendors forget to close the fuse :(
- Relies on a chain of trust based on Intel BIOS Authenticated Code Modules (ACM).

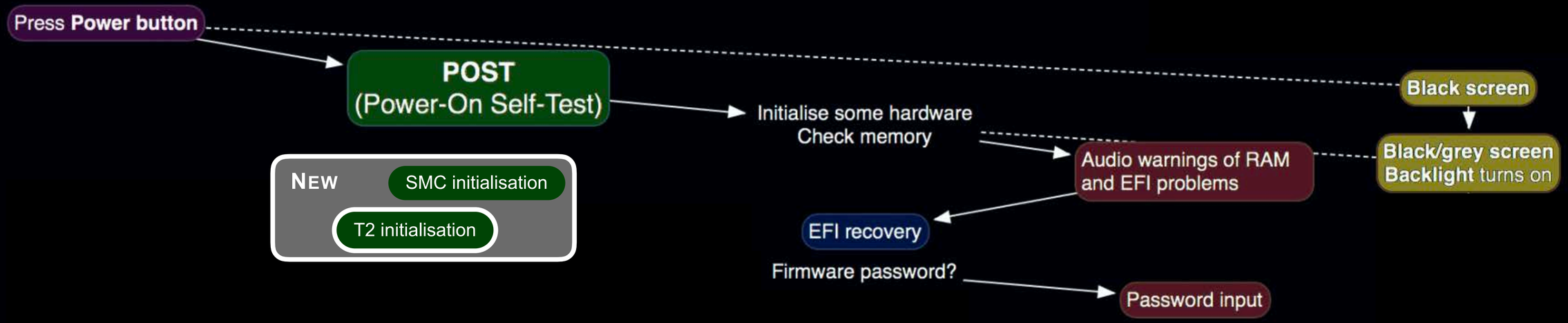
Alex Ermolov @ ZerNights - https://2016.zeronights.ru/wp-content/uploads/2017/03/Intel-BootGuard.pdf
https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/security-technologies-4th-gen-core-retail-paper.pdf
https://en.wikipedia.org/wiki/Trusted_Execution_Technology#Chain_of_trust


32.768k
GLF
T2
PGMU6400LL 1744

Apple Systems shipped with T2

- iMac Pro 2017
  released 14/12/2017
- MacBook Pro 2018
  released 12/07/2018
- MacBook Air
  released 07/11/2018
- Mac mini
  released 07/11/2018

Secure Boot on the Mac



Apple T2 and Secure Storage

"Data on your iMac Pro built-in, solid-state drive (SSD) is encrypted using a hardware accelerated AES engine built into the T2 chip. The encryption is performed with 256 bit keys tied to a unique identifier within the T2 chip."

Apple 'About encrypted storage on your new Mac' https://support.apple.com/en-us/HT208344

- T2 chip connects SSD to CPU via Non-Volatile Memory Express (NVMe)

Apple T2 and Secure Storage

"Data on your iMac Pro built-in, solid-state drive (SSD) is encrypted using a hardware accelerated AES engine built into the T2 chip. The encryption is performed with 256 bit keys tied to a unique identifier within the T2 chip."

Apple 'About encrypted storage on your new Mac' https://support.apple.com/en-us/HT208344

- T2 chip connects SSD to CPU via Non-Volatile Memory Express (NVMe)

Apple T2 and Secure Storage

"Data on your iMac Pro built-in solid-state drive (SSD) is encrypted using a hardware ac d with

ple.com/en-us/HT208344

- T2 chip

```
$ ioreg -p IODeviceTree -w 0 -l
<condensed output follows>
PCI0@0
(IOACPIFamily.kext)
    RP01@1C
    (IOPCIFamily.kext)
        ANS2@0
        (IOPCIFamily.kext)
            AppleANS2Controller
            (IONVMeFamily.kext)
                IONVMeBlockStorageDevice@1
                (IONVMeFamily.kext)
                    IOBlockStorageDriver
                    (IOStorageFamily.kext)
                        APPLE SSD AP1024M Media
```

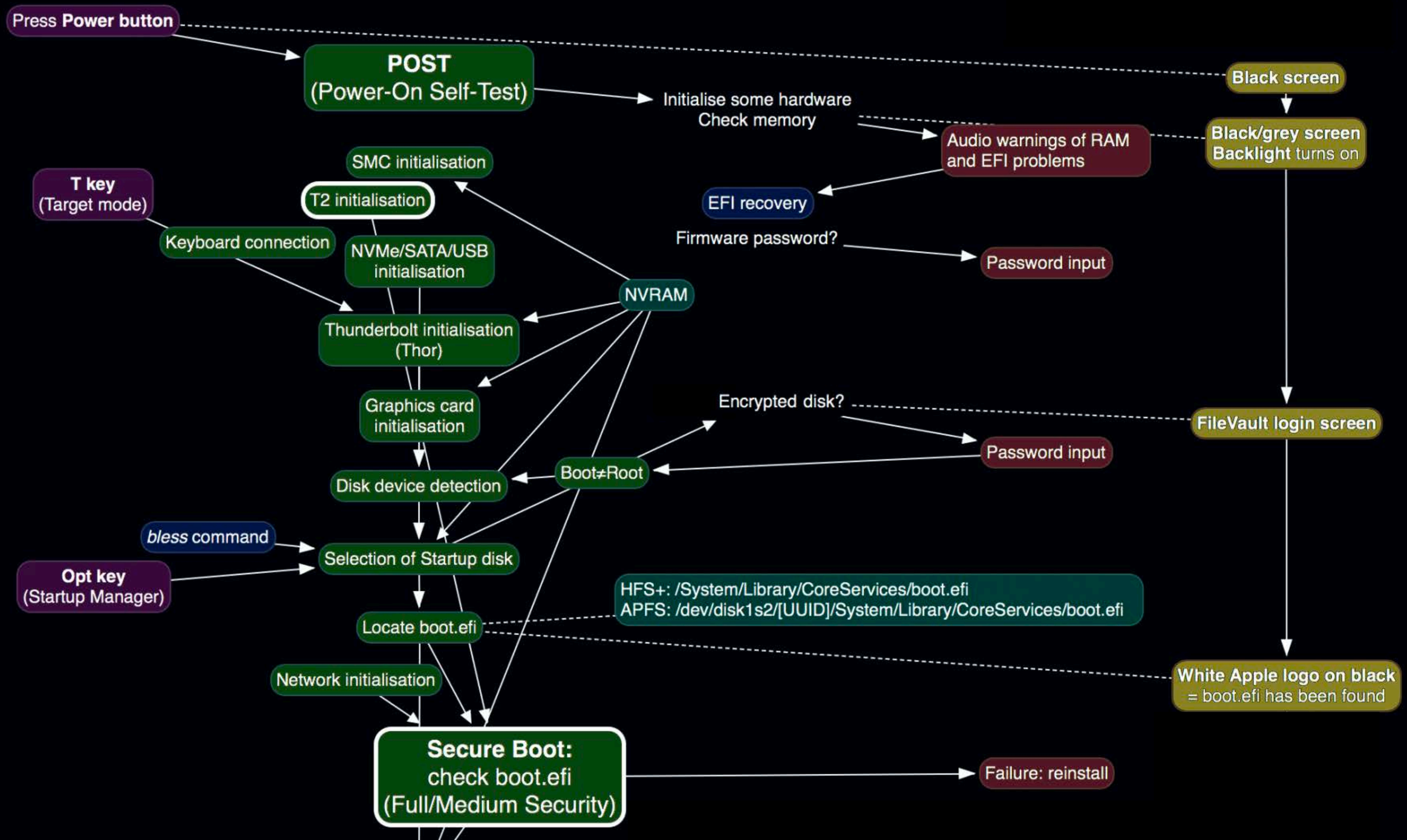
Apple T2 and Secure Storage

"Data on your iMac Pro built-in, solid-state drive (SSD) is encrypted using a hardware accelerated AES engine built into the T2 chip. The encryption is performed with 256 bit keys tied to a unique identifier within the T2 chip."

Apple 'About encrypted storage on your new Mac' https://support.apple.com/en-us/HT208344

- T2 chip connects SSD to CPU via Non-Volatile Memory Express (NVMe)
- T2 coprocessor and the SSD chips are uniquely bound together (fused)
- T2 chip does hardware-accelerated AES block level encryption/decryption using XTS-AES and 128 bit blocks with a 256-bit key

- Pepijn Bruienne's article 'Apple iMac Pro and Secure Storage' *https://duo.com/blog/apple-imac-pro-and-secure-storage*
- Apple T2 white paper *https://www.apple.com/mac/docs/Apple_T2_Security_Chip_Overview.pdf*


Press **Power button**
POST
(Power-On Self-Test)
Initialise some hardware
Check memory
Black screen
Audio warnings of RAM and EFI problems
Black/grey screen
Backlight turns on
EFI recovery
SMC initialisation
T2 initialisation
T key
(Target mode)
Keyboard connection
NVMe/SATA/USB initialisation
Firmware password?
Password input
NVRAM
Thunderbolt initialisation
(Thor)
Graphics card initialisation
Encrypted disk?
FileVault login screen
Password input
Boot≠Root
Disk device detection
bless command
Opt key
(Startup Manager)
Selection of Startup disk
HFS+: /System/Library/CoreServices/boot.efi
APFS: /dev/disk1s2/[UUID]/System/Library/CoreServices/boot.efi
Locate boot.efi
Network initialisation
White Apple logo on black
= boot.efi has been found
Secure Boot:
check boot.efi
(Full/Medium Security)
Failure: reinstall

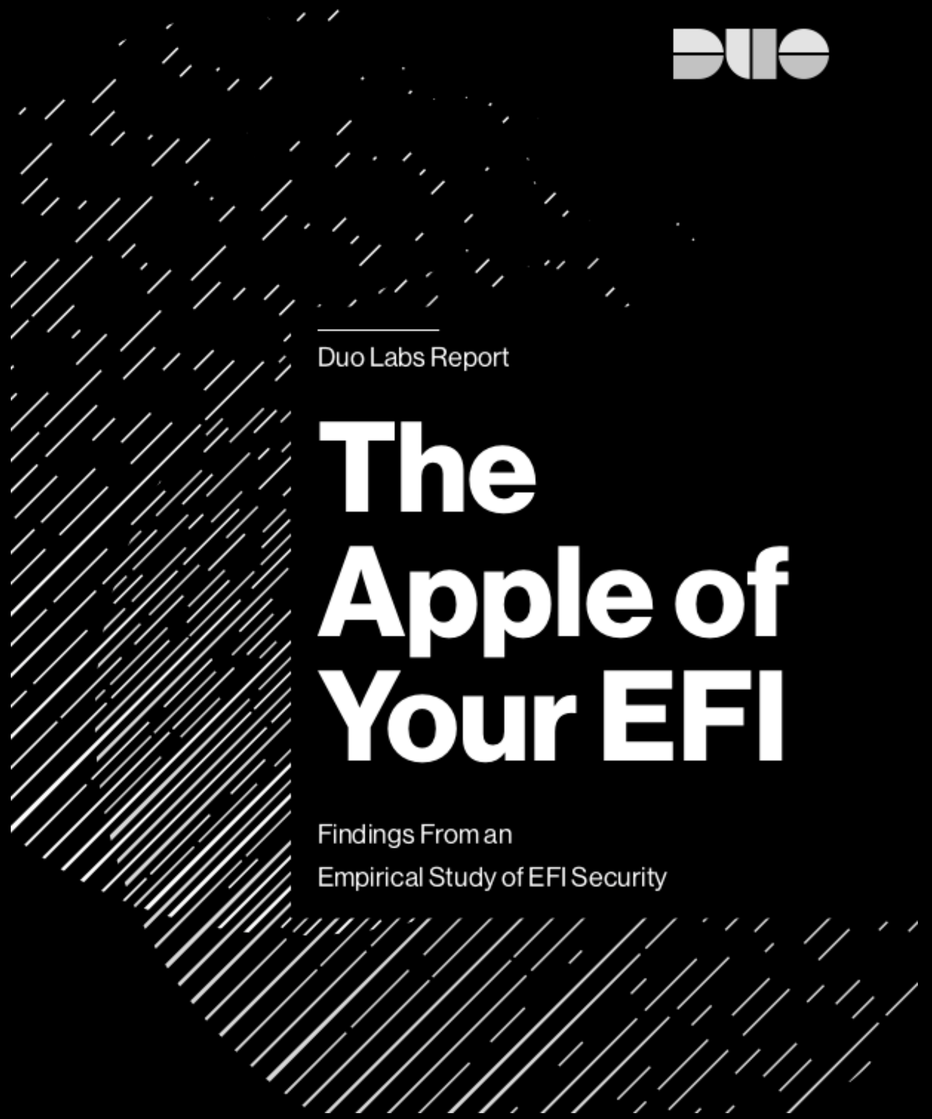
Boot Device Selection

- Traditionally using `bless` command to
  - declare volume as bootable in the volume header,
  - sett NVRAM variables efi-boot-device and efi-boot-device-data,
  - copy boot.efi to the right place, and/or
  - configure the Mac to boot using NetBoot.
- System Image Protection introduced `systemsetup -setstartupdisk` but `sudo bless --getBoot --verbose` still works.
- APFS introduced the APFS Preboot volume for the initial boot stage `diskutil apfs list`
- APFS introduced a root folder structure based on GUIDs per boot volume

Unified Extensible Firmware Interface (UEFI)

- boot.efi software (“OS X booter”)
  - system specific
  - replacing BootX
  - confusingly referred to by Apple as BootROM
  - config `/Library/Preferences/SystemConfiguration/com.apple.Boot.plist`
- Main tasks
  - built the device tree, IODeviceTree, which lists all the devices in that Mac
  - "snag keys" – startup key commands
  - display login UI for FileVault2
  - gather kernel options and parameters
  - check kernel cache (unified prelinked kernel OS X 10.7+) *or identify and link (slow)*
  - start kernel
  - kernel initializes the Mach and BSD data structures and then initializes the I/O Kit.

https://developer.apple.com/library/archive/documentation/Darwin/Conceptual/KernelProgramming/booting/booting.html



The Apple of Your EFI

Findings From an Empirical Study of EFI Security

AUTHORS
Rich Smith
Pepijn Bruienne

EDITOR
Thu T. Pham

DESIGNER
Chelsea Lewis

VERSION
1.1

Table of Contents

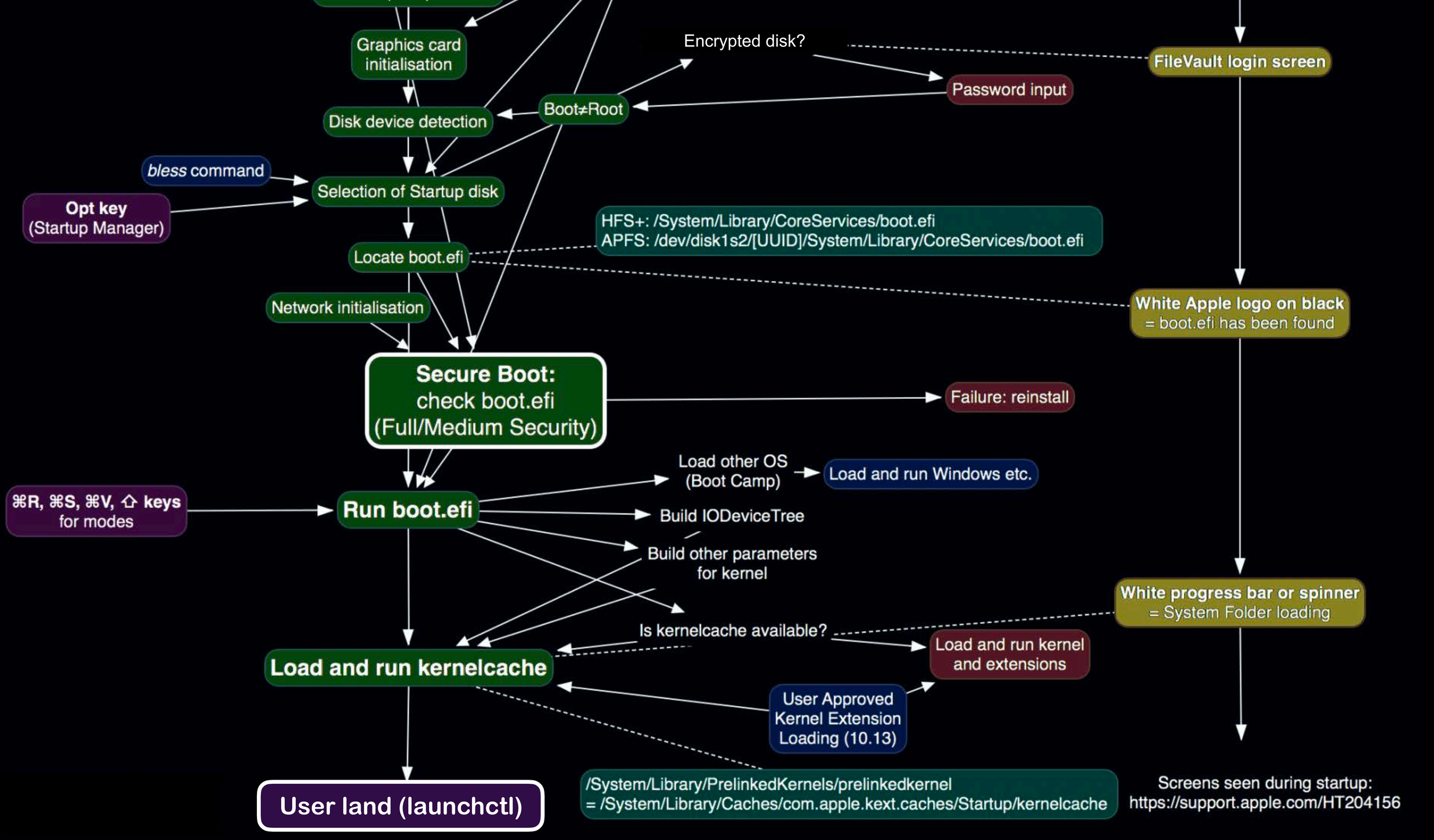
https://duo.com/assets/ebooks/Duo-Labs-The-Apple-of-Your-EFI.pdf

Apple T2 and Secure Boot implemented in EFI

The T2 chip and SecureBoot add more complexity to the PreBoot volume:

- Certificates and signatures for OS components stored in `.im4m` file format (use openssl asn1parse).
- Medium security operates with OS specific but system generic `.im4m` files, e.g. `boot.efi.j137ap.im4m`
- Full security mode requires the `.im4m` to be system specific based on a unique Exclusive Chip Identification (ECID), e.g. `boot.efi.j137ap.1234567890ABCD.im4m`
- This uses a new personalisation process using the Signed Hash protocol (SHSH) following the known process from iOS to validate firmware signatures.

http://michaellynn.github.io/2018/07/27/booting-secure/
https://www.theiphonewiki.com/wiki/ECID
https://www.theiphonewiki.com/wiki/SHSH


Graphics card initialisation
Encrypted disk?
FileVault login screen
Password input
Boot≠Root
Disk device detection
bless command
Selection of Startup disk
Opt key
(Startup Manager)
HFS+: /System/Library/CoreServices/boot.efi
APFS: /dev/disk1s2/[UUID]/System/Library/CoreServices/boot.efi
Locate boot.efi
White Apple logo on black
= boot.efi has been found
Network initialisation
Secure Boot:
check boot.efi
(Full/Medium Security)
Failure: reinstall
Load other OS
(Boot Camp)
Load and run Windows etc.
⌘R, ⌘S, ⌘V, ⇧ keys
for modes
Run boot.efi
Build IODeviceTree
Build other parameters
for kernel
White progress bar or spinner
= System Folder loading
Is kernelcache available?
Load and run kernel
and extensions
Load and run kernelcache
User Approved
Kernel Extension
Loading (10.13)
User land (launchctl)
/System/Library/PrelinkedKernels/prelinkedkernel
= /System/Library/Caches/com.apple.kext.caches/Startup/kernelcache
Screens seen during startup:
https://support.apple.com/HT204156


Finder File Edit View Go Window Help
Mon 9:41 AM
MacBook Pro

Thank You!

https://github.com/mjung/publications

Marko Jung

Global Head of Information Security Operations

m@mju.ng @mjung fb.com/markohjung